PROCESSOR

Patent Number: JP5224927
Publication date: 1993-09-03
Inventor(s): KIMURA KOZO; others: 02
Applicant(s):: MATSUSHITA ELECTRIC IND CO LTD
Requested Patent: JP5224927
Application Number: JP19920305700 19921116
Priority Number(s):
IPC Classification: G06F9/38
EC Classification:
Equivalents: JP3146077B2

Abstract

PURPOSE:To provide the processor which processes speculatively an instruction, while a condition corresponding to a condition branch instruction is not defined, scarcely has an interlock, and executes a program at a high speed.
CONSTITUTION:An instruction decoding part 7 discriminates the kind of a condition branch instruction contained in an instruction train before its execution, and in accordance with the kind of the discriminated instruction, with respect to arithmetic units 8–11, an instruction of an instruction train of a branch destination and/or a succeeding instruction train is issued in parallel as a speculative execution to the executing unit. A result of the speculative execution by the arithmetic units 8–11 is stored temporarily in an arithmetic unit management table 12. When success or failure of a branch of the condition branch instruction is decided by an instruction execution matching maintaining circuit 18, an execution sequence managing circuit 17 discriminates whether a result of execution of the instruction train is valid or invalid, stores a valid execution result in a register file 19, and on the other hand, erases an invalid execution result from an execution sequence management buffer 13.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平5－224927

(43)公開日　平成5年（1993）9月3日

(51)Int.Cl.[5]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　9/38	３３０　Ｋ	9290－5Ｂ		

審査請求　未請求　請求項の数16（全 29 頁）

(21)出願番号　特願平4－305700

(22)出願日　平成4年（1992）11月16日

(31)優先権主張番号　特願平３－３００１２６

(32)優先日　平3（1991）11月15日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005821
松下電器産業株式会社
大阪府門真市大字門真1006番地

(72)発明者　木村　浩三
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　岡　康介
大阪府門真市大字門真1006番地　松下電器産業株式会社内

(72)発明者　清原　督三
大阪府門真市大字門真1006番地　松下電器産業株式会社内

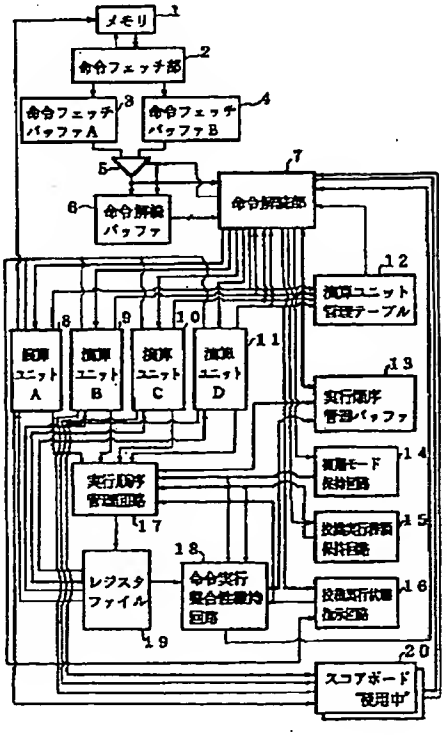
(74)代理人　弁理士　中島　司朗

(54)【発明の名称】プロセッサ

(57)【要約】

【目的】　本発明は、条件分岐命令に対応する条件が確定していない間、投機的に命令を処理し、インターロックが少なく、プログラムを高速に実行するプロセッサを提供することを目的とする。

【構成】　命令解読部７が実行前の命令列に含まれる条件分岐命令の種類を判別し、判別された命令の種類に応じて、演算ユニット８～１１に対して、分岐先の命令列及び／又は後続する命令列の命令を実行ユニットに対して投機実行として並列発行する。演算ユニット８～１１にて陶器実行された結果は、演算ユニット管理テーブル１２に一時的に格納される。命令実行整合性維持回路１８によって条件分岐命令の分岐の成否が判定されると、実行順序管理回路１７は、命令列の実行結果の有効無効を識別し、有効な実行結果をレジスタファイル１９に格納する一方、無効な実行結果を実行順序管理バッファ１３から消去する。



1

【特許請求の範囲】

【請求項１】複数の実行ユニットを有し、メモリにある命令列の命令を並列に処理するプロッセサであって、

実行前の命令列に含まれ、条件が他の命令に依存する条件分岐命令の種類を判別する命令種別判別手段と、

分岐の成否が決定されるまでの間、条件分岐命令の種類に応じて、実行ユニットに対して、分岐先の命令列及び／又は後続する命令列の命令を実行ユニットに対して並列発行する命令並列発行手段と、

前記条件分岐が依存する他の命令が実行されたとき、条件分岐命令の分岐の成否を判定する分岐判定手段と、

条件分岐命令の分岐の成否の判定結果によって、命令列の実行結果の有効無効を識別する実行結果管理手段とを備えたことを特徴とするプロセッサ。

【請求項２】前記命令種別判別手段は、

分岐する確率が分岐しない確率と同程度の第１の種類の命令と、　それ以外の分岐命令とを判別し、

前記命令並列発行手段は、条件分岐命令が第１の種類の命令であった場合、分岐先の命令列と条件分岐命令に後続する命令列の双方を実行ユニットに対して並列発行し、条件分岐命令が第１の種類の命令以外の命令であった場合、どちらか一方を実行ユニットに対して並列発行し、

前記実行結果管理手段は、条件分岐命令が第１の種類の命令であった場合、分岐の成否の判定結果によって、一方の命令列の実行結果を有効にし、他方を無効にする一方、条件分岐命令が第１の種類の命令以外の場合、分岐の成否の判定結果に応じて命令列の実行結果の全てを有効又は無効にすることを特徴とする請求項１記載のプロセッサ。

【請求項３】前記命令種別判別手段は、第１の分岐命令以外の命令を、第１の分岐命令より分岐する確率が高い第２の種類の命令と、第１の分岐命令より分岐しない確率が高い第３の種類の命令とに判別し、

前記命令並列発行手段は、第２の種類の命令の場合、分岐先の命令列の命令を実行ユニットに対して並列発行し、第３の種類の命令の場合、条件分岐命令に後続する命令列の命令を実行ユニットに並列発行することを特徴とする請求項２記載のプロセッサ。

【請求項４】前記のプロセッサは、さらに条件分岐命令に後続する命令列の命令を一時的に記憶する第１の命令フェッチバッファと、

条件分岐命令の分岐先の命令列の命令を一時的に記憶する第２の命令フェッチバッファと、

メモリに格納された命令を読み出し、第１及び第２の命令フェッチバッファに格納する命令フェッチ手段とを備えたことを特徴とする請求項１記載のプロセッサ。

【請求項５】前記命令フェッチ手段は、

メモリに読み出しアドレスを出力し、その内容をインクリメントするプログラムカウンタと、

2

メモリから読みだされた命令から条件分岐命令を検出する条件分岐命令検出手段と、

条件分岐命令検出手段で検出された条件分岐命令に基づいて、分岐先のアドレスを計算する演算手段と、

通常、メモリから命令が読みだされるごとにプログラムカウンタの内容をインクリメントしていき第１の命令フェッチバッファに命令を格納し、条件分岐命令が検出された場合、後続する命令列の命令を第１の命令フェッチバッファに格納し、演算手段で求められた分岐先アドレスをプログラムカウンタに書き込み、分岐先の命令列の命令を第２の命令フェッチバッファに格納するフェッチ制御手段とを有することを特徴とする請求項４記載のプロセッサ。

【請求項６】前記命令並列発行手段は、

命令を解読して実行ユニットへの制御信号を生成する複数の解読手段と、

第１又は、第２の命令フェッチバッファから全ての解読手段に命令を１つずつ転送する転送制御手段と、

転送制御手段により転送されるそれぞれの命令に、その命令がどの命令列に属するかを示すモードを付加し、分岐命令に後続する命令列と分岐先の命令列とでモードを変更するモード付加手段とを有することを特徴とする請求項４記載のプロセッサ。

【請求項７】前記モード付加手段が付加するモードは、２ビットの情報であり、条件分岐命令に後続する命令列のモードは条件分岐命令自身のモードに対して第１の値を加算した値、分岐先の命令列のモードは第２の値を加算した値であることを特徴とする請求項６記載のプロセッサ。

【請求項８】前記条件分岐命令が依存する他の命令は、ＰＳＲ（Program Status Register）を変更する命令であり、

前記分岐判定手段は、ＰＳＲを変更する命令が実行されたことを検出し、その実行結果を参照することを特徴とする請求項６記載のプロセッサ。

【請求項９】前記プロセッサは、さらに、

解読手段で解読された複数の解読結果を参照して、解読された複数の命令間のデータ依存関係を判定するデータ依存関係判定手段と、

実行ユニットで実行中の命令が使用しているレジスタがどれであるかを管理するスコアボード管理手段と、

実行ユニットの空き状態を検知する空き状態検知手段と、

解読手段で解読された各命令について、データ依存関係判定手段の判定結果、スコアボード管理手段、空き状態検知手段の検知結果に基づいて、発行可能な命令を命令並列発行手段から実行ユニットへの発行を許可し、また、解読手段で解読された命令が条件分岐命令である場合には、命令並列発行手段から分岐判定手段への発行を許可する命令発行許可手段とを有することを特徴とする

請求項6記載のプロセッサ。

【請求項10】前記命令発行許可手段は、

複数の解読手段の解読結果にそれぞれについて、データ依存関係検出手段の検出結果から前の命令とデータ依存関係がなく、スコアボード管理手段の情報からその命令のオペランドで指定されているレジスタが実行ユニットで実行中の命令に使用されていなく、かつ、空き状態検知手段の検知結果からその命令を実行しうる実行ユニットが空いていること、を満たす命令を発行可能と判定することを特徴とする請求項9記載のプロセッサ。

【請求項11】前記プロセッサはさらに、

実行ユニット毎に命令を実行中であるか否かを示す情報を保持し、空き状態検知手段によって参照される実行ユニット管理テーブルと、

実行ユニットで実行中の命令が使用しているレジスタがどれであるかを示す情報を保持し、スコアボード管理手段によって参照されるスコアボードとを備えたことを特徴とする請求項9記載のプロセッサ。

【請求項12】前記実行結果管理手段は、

命令並列発行手段によって発行された条件分岐命令のモードを初期モードとして保持する初期モード保持手段と、

命令並列発行手段によって発行された条件分岐命令の種類を保持する投機実行種類保持手段と、

命令並列発行手段によって発行された条件分岐命令の条件判断が未だ確定していないことを示すフラグを保持する投機実行状態指示手段とを有し、

前記命令並列発行手段は、条件分岐命令を分岐判定手段に発行すると同時に、その条件分岐命令のモードを初期モード保持手段に、その条件分岐命令の種類を投機実行種類保持手段に出力し、投機実行状態指示手段のフラグをセットすることを特徴とする請求項9記載のプロセッサ。

【請求項13】前記は、実行結果管理手段は、

実行ユニットでの命令の実行結果と、その命令のモードと、その実行結果が本来格納されるべき格納先の情報と対応させて記憶する一時記憶手段を有することを特徴とする請求項12記載のプロセッサ。

【請求項14】前記一時記憶手段は、

格納先の情報として、本来の格納先がレジスタである場合には、レジスタ番号を、本来の格納先がメモリである場合には、メモリアドレスを記憶することを特徴とする請求項13記載のプロセッサ。

【請求項15】分岐判定手段は、分岐するか否かを判定すると投機実行状態指示手段のフラグをクリアし、

実行結果管理手段は、分岐判定手段の判定結果に基づいて、初期モード保持手段、投機実行種類保持手段を参照して一時記憶手段の実行結果の有効無効を識別し、有効な実行結果を本来の格納先に転送し、無効な実行結果をクリアすることを特徴とする請求項13記載のプロセッサ。

【請求項16】命令フェッチ手段は、

条件分岐命令に後続する命令列の命令を一時的に格納する第1の命令フェッチバッファと、

条件分岐命令の分岐先の命令を一時的に格納する第2の命令フェッチバッファと、

メモリに読み出しアドレスを出力し、その内容をインクリメントするプログラムカウンタと、

メモリから読みだされた命令から条件分岐命令を検出する条件分岐命令検出手段と、

条件分岐命令検出手段で検出された条件分岐命令に基づいて、分岐先のアドレスを計算する演算手段と、

通常、メモリから命令が読みだされるごとにプログラムカウンタの内容をインクリメントしていき第1の命令フェッチバッファに命令を格納し、条件分岐命令が検出された場合、後続する命令列の命令を第1の命令フェッチバッファに格納し、演算手段で求められた分岐先アドレスをプログラムカウンタに書き込み、分岐先の命令列の命令を第2の命令フェッチバッファに格納するフェッチ制御手段とを有し、

命令並列発行手段は、

命令を解読して実行ユニットへの制御信号を生成する複数の解読手段と、

第1又は、第2の命令フェッチバッファから全ての解読手段に命令を1つずつ転送する転送制御手段と、

転送制御手段により転送されるそれぞれの命令に、その命令がどの命令列に属するかを示すモードを付加し、分岐命令に後続する命令列と分岐先の命令列とでモードを変更するモード付加手段とを有し、

プロセッサは、

解読手段で解読された複数の解読結果を参照して、解読された複数の命令間のデータ依存関係を判定するデータ依存関係判定手段と、

実行ユニットで実行中の命令が使用しているレジスタがどれであるかを管理するスコアボード管理手段と、

実行ユニットの空き状態を検知する空き状態検知手段と、

解読手段で解読された各命令について、データ依存関係判定手段の判定結果、スコアボード管理手段、空き状態検知手段の検知結果に基づいて、発行可能な命令を命令並列発行手段から実行ユニットへの発行を許可し、また、解読手段で解読された命令が条件分岐命令である場合には、命令並列発行手段から分岐判定手段への発行を許可する命令発行許可手段とを有し、

実行結果管理手段は、

実行ユニットでの命令の実行結果と、その命令のモードと、その実行結果が本来格納されるべき格納先の情報と対応させて記憶する一時記憶手段と、

命令並列発行手段によって発行された条件分岐命令のモードを初期モードとして保持する初期モード保持手段

と、
命令並列発行手段によって発行された条件分岐命令の種類を保持する投機実行種類保持手段と、
命令並列発行手段によって発行された条件分岐命令の条件判断が未だ確定していないことを示すフラグを保持する投機実行状態指示手段とを有し、
前記命令並列発行手段は、条件分岐命令を分岐判定手段に発行すると同時に、その条件分岐命令のモードを初期モード保持手段に、その条件分岐命令の種類を投機実行種類保持手段に出力し、投機実行状態指示手段のフラグをセットし、
実行結果管理手段は、分岐判定手段の判定結果を受けると、
投機実行種類保持手段に第１の種類の命令が保持されている場合で、かつ、分岐すると判定された場合、初期モードを参照して一時記憶手段から後続する命令列の実行結果をクリアして分岐先の命令列の実行結果を本来の格納先へ転送し、逆に、分岐しないと判定された場合、初期モードを参照して一時記憶手段から分岐先の命令列の実行結果をクリアして後続する命令列の実行結果を本来の格納先へ転送し、
投機実行種類保持手段に第２の種類の命令が保持されている場合で、かつ、分岐すると判定された場合、初期モードを参照して一時記憶手段から分岐先の命令列の実行結果を本来の格納先へ転送し、逆に、分岐しないと判定された場合、一時記憶手段から分岐先の命令列の実行結果をクリアし、
投機実行種類保持手段に第３の種類の命令が保持されている場合で、かつ、分岐すると判定された場合、初期モードを参照して一時記憶手段から後続する命令列の実行結果を本来の格納先へ転送し、逆に、分岐しないと判定された場合、一時記憶手段から後続する命令列の実行結果をクリアする。ことを特徴とする請求項３記載のプロセッサ。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、並列実行可能な複数の演算処理ユニットを有するプロセッサにおける条件分岐命令の高速処理に関する。

【０００２】

【従来技術】近年、プロセッサにおける高性能化は、SuperscalarやVLIWと呼ばれる複数の演算ユニットを用意し、一度に複数命令を並列に実行する方式が採られるようになってきた。これらの方式で商用化されたものには、MOTOROLA社のマイクロプロセッサMC88100等が知られている（MOTOROLA社の「MC88100 MICROPROCESSOR USER'S MANUAL SECOND EDITION」に示されている）。これらの方式を用いると、従来シーケンシャルに実行されていた命令列が、複数の命令を一度に実行することが可能となる。

【０００３】また、命令の実行順序に関して、命令列の並ぶ順に実行するのではなく、データの依存関係がない命令については実行可能なものから実行する、すなわちout-of-order実行という高速化方法も知られている。これらの高速化技法については信学技報CPSY-90-54（'90.7）「SIMP（単一命令流／多重命令パイプライン）方式に基づくスーパースカラ・プロセッサの改良方針」（「An Extended Superscalar Processor Prototype Based on the SIMP (Sngle Instruction stream/Multiple instruction Pipelining) Architecture」）に示されている。

【０００４】上記従来技術のプロセッサは、条件分岐命令の分岐が確定していなくても、どちらに分岐するかを予測して、先行的に命令を実行（投機的実行と呼ぶ）することにより、高速化を図っていた。予測の方法は、例えば、以前出現したものと同じ条件分岐命令は前回と同じ側に分岐すると予測する、などがある。

【０００５】

【発明が解決しようとする課題】しかしながら、上記のような従来技術によれば、以下のような問題点を有していた。一般に、複数の命令が同時に実行されるようになると、処理サイクルあたりの分岐命令の出現頻度が増える。条件分岐命令の場合には、条件が確定（通常はＰＳＲ：Program Status Registerの変更）するまで分岐するか否かは決定できない。このため条件分岐命令毎に命令流がインターロックされてしまい、性能が頭打ちの状態になる。（これら分岐による依存関係を制御依存と呼ぶ）

また、分岐するかしないかを予測して、先行的に命令を処理（投機的実行と呼ぶ）する方法では、個々の条件分岐命令に関しては、予測が当たった場合は高速化が達成されるが、予測が外れた場合はもう一方の命令列に戻って命令の実行をやり直すことになり、その代償が大きい。さらに、プログラム全体の実行に関しては、必ずしも高速化が達成されるとは限らない。

【０００６】本発明の目的は、条件分岐命令に対応する条件が確定していなくても投機的に命令を処理し、インターロックが少なく、プログラムを高速に実行するプロセッサを提供することにある。

【０００７】

【課題を解決するための手段】上記の課題を解決するため、本発明のプロセッサは、複数の実行ユニットを有し、メモリにある命令列の命令を並列に処理するプロッセサであって、実行前の命令列に含まれ、条件が他の命令に依存する条件分岐命令の種別を判別する命令種別判別手段と、分岐の成否が決定されるまでの間、条件分岐命令の種類に応じて、実行ユニットに対して、分岐先の命令列及び／又は後続する命令列の命令を実行ユニットに対して並列発行する命令並列発行手段と、前記条件分岐が依存する他の命令が実行されたとき、条件分岐命令

の分岐の成否を判定する分岐判定手段と、条件分岐命令の分岐の成否の判定結果によって、命令列の実行結果の有効無効を識別する実行結果管理手段とを備えている。

【０００８】前記命令種別判別手段は、分岐する確率が分岐しない確率と同程度の第１の種類の命令と、　それ以外の分岐命令とを判別し、前記命令発行並列手段は、条件分岐命令が第１の種類の命令であった場合、分岐先の命令列と条件分岐命令に後続する命令列の双方を実行ユニットに対して並列発行し、条件分岐命令が第１の種類の命令以外の命令であった場合、どちらか一方を実行ユニットに対して並列発行し、前記実行結果管理手段は、条件分岐命令が第１の種類の命令であった場合、分岐の成否の判定結果によって、一方の命令列の実行結果を有効にし、他方を無効にする一方、条件分岐命令が第１の種類の命令以外の場合、分岐の成否の判定結果に応じて命令列の実行結果の全てを有効又は無効にするような構成であってもよい。

【０００９】前記命令種別判別手段は、第１の分岐命令以外の命令を、第１の分岐命令より分岐する確率が高い第２の種類の命令と、第１の分岐命令より分岐しない確率が高い第３の種類の命令とに判別し、前記命令並列発行手段は、第２の種類の命令の場合、分岐先の命令列の命令を実行ユニットに対して並列発行し、第３の種類の命令の場合、条件分岐命令に後続する命令列の命令を実行ユニットに並列発行するようなこうせいであってもよい。

【００１０】前記のプロセッサは、さらに条件分岐命令に後続する命令列の命令を一時的に記憶する第１の命令フェッチバッファと、条件分岐命令の分岐先の命令列の命令を一時的に記憶する第２の命令フェッチバッファと、メモリに格納された命令を読み出し、第１及び第２の命令フェッチバッファに格納する命令フェッチ手段とを備えていてもよい。

【００１１】前記命令フェッチ手段は、メモリに読み出しアドレスを出力し、その内容をインクリメントするプログラムカウンタと、メモリから読みだされた命令から条件分岐命令を検出する条件分岐命令検出手段と、条件分岐命令検出手段で検出された条件分岐命令に基づいて、分岐先のアドレスを計算する演算手段と、通常、メモリから命令が読みだされるごとにプログラムカウンタの内容をインクリメントしていき第１の命令フェッチバッファに命令を格納し、条件分岐命令が検出された場合、後続する命令列の命令を第１の命令フェッチバッファに格納し、演算手段で求められた分岐先アドレスをプログラムカウンタに書き込み、分岐先の命令列の命令を第２の命令フェッチバッファに格納するフェッチ制御手段とを有していてもよい。

【００１２】前記命令並列発行手段は、命令を解読して実行ユニットへの制御信号を生成する複数の解読手段と、第１又は、第２の命令フェッチバッファから全ての解読手段に命令を１つずつ転送する転送制御手段と、転送制御手段により転送されるそれぞれの命令に、その命令がどの命令列に属するかを示すモードを付加し、分岐命令に後続する命令列と分岐先の命令列とでモードを変更するモード付加手段とを有していてもよい。

【００１３】前記モード付加手段が付加するモードは、２ビットの情報であり、条件分岐命令に後続する命令列のモードは条件分岐命令自身のモードに対して第１の値を加算した値、分岐先の命令列のモードは第２の値を加算した値であってもよい。前記条件分岐命令が依存する他の命令は、ＰＳＲ（Program Status Register）を変更する命令であり、前記分岐判定手段は、ＰＳＲを変更する命令が実行されたことを検出し、その実行結果を参照するような構成であってもよい。

【００１４】前記プロセッサは、さらに、解読手段で解読された複数の解読結果を参照して、解読された複数の命令間のデータ依存関係を判定するデータ依存関係判定手段と、実行ユニットで実行中の命令が使用しているレジスタがどれであるかを管理するスコアボード管理手段と、実行ユニットの空き状態を検知する空き状態検知手段と、解読手段で解読された各命令について、データ依存関係判定手段の判定結果、スコアボード管理手段、空き状態検知手段の検知結果に基づいて、発行可能な命令を命令並列発行手段から実行ユニットへの発行を許可し、また、解読手段で解読された命令が条件分岐命令である場合には、命令並列発行手段から分岐判定手段への発行を許可する命令発行許可手段とを有していてもよい。

【００１５】前記命令発行許可手段は、複数の解読手段の解読結果にそれぞれについて、データ依存関係検出手段の検出結果から前の命令とデータ依存関係がなく、スコアボード管理手段の情報からその命令のオペランドで指定されているレジスタが実行ユニットで実行中の命令に使用されていなく、かつ、空き状態検知手段の検知結果からその命令を実行しうる実行ユニットが空いていること、を満たす命令を発行可能と判定することを特徴とする請求項９記載のプロセッサ。

【００１６】前記プロセッサはさらに、実行ユニット毎に命令を実行中であるか否かを示す情報を保持し、空き状態検知手段によって参照される実行ユニット管理テーブルと、実行ユニットで実行中の命令が使用しているレジスタがどれであるかを示す情報を保持し、スコアボード管理手段によって参照されるスコアボードとを備えていてもよい。

【００１７】前記実行結果管理手段は、命令並列発行手段によって発行された条件分岐命令のモードを初期モードとして保持する初期モード保持手段と、命令並列発行手段によって発行された条件分岐命令の種類を保持する投機実行種類保持手段と、命令並列発行手段によって発行された条件分岐命令の条件判断が未だ確定していない

ことを示すフラグを保持する投機実行状態指示手段とを有し、前記命令並列発行手段は、条件分岐命令を分岐判定手段に発行すると同時に、その条件分岐命令のモードを初期モード保持手段に、その条件分岐命令の種類を投機実行種類保持手段に出力し、投機実行状態指示手段のフラグをセットするような構成であってもよい。

【００１８】前記は、実行結果管理手段は、実行ユニットでの命令の実行結果と、その命令のモードと、その実行結果が本来格納されるべき格納先の情報と対応させて記憶する一時記憶手段を有していてもよい。前記一時記憶手段は、格納先の情報として、本来の格納先がレジスタである場合には、レジスタ番号を、本来の格納先がメモリである場合には、メモリアドレスを記憶するような構成であってもよい。

【００１９】分岐判定手段は、分岐するか否かを判定すると投機実行状態指示手段のフラグをクリアし、実行結果管理手段は、分岐判定手段の判定結果に基づいて、初期モード保持手段、投機実行種類保持手段を参照して一時記憶手段の実行結果の有効無効を識別し、有効な実行結果を本来の格納先に転送し、無効な実行結果をクリアするような構成であってもよい。

【００２０】

【作用】上記の手段により本発明のプロセッサは、命令種別判別手段が実行前の命令列に含まれる条件分岐命令の種類を判別する。この判別された命令の種類に応じて、命令並列発行手段は、実行ユニットに対して、分岐先の命令列及び／又は後続する命令列の命令を実行ユニットに対して並列発行する。分岐判定手段によって、条件分岐命令の分岐の成否が判定されると、実行結果管理手段は、命令列の実行結果の有効無効を識別する。

【００２１】さらに、命令種別管理手段は、分岐する確率が分岐しない確率と同程度の第１の種類の命令と、それ以外の命令を判別する。それ以外の命令について、第１の種類の分岐命令より分岐する確率が高い第２の種類の分岐命令と、第１の種類の分岐命令より分岐しない確率が高い第３の種類の分岐命令とを判別する。命令フェッチ手段は、通常メモリから第１の命令フェッチ手段に命令を格納していき、条件分岐命令検出手段によって条件分岐命令が検出された場合、条件分岐命令に後続する命令列の命令を第１の命令フェッチバッファに、分岐先の命令列の命令を第２の命令フェッチバッファに格納する。これらの命令フェッチバッファに格納された命令について、命令並列発行手段は、モード付加手段によりモードを付加しつつ、転送制御手段によって２つの命令フェッチバッファから複数の解読手段に対して命令を取り込む。

【００２２】さらに、命令発行許可手段は、複数の解読手段で解読された各命令について、データ依存関係判定手段の判定結果、スコアボード管理手段、空き状態検知手段の検知結果に基づいて、発行可能な命令を命令並列

発行手段から実行ユニットへの発行を許可し、また、解読手段で解読された命令が条件分岐命令である場合には、命令並列発行手段から分岐判定手段への発行を許可する。このとき、命令発行許可手段は、条件分岐命令が第１の種類の命令であった場合、分岐先の命令列と条件分岐命令に後続する命令列の双方を実行ユニットに対して並列発行を許可し、第２の種類の命令の場合、分岐先の命令列の命令を実行ユニットに対して並列発行し、第３の種類の命令の場合、条件分岐命令に後続する命令列の命令を実行ユニットに並列発行を許可する。

【００２３】前記命令並列発行手段は、条件分岐命令を分岐判定手段に発行すると同時に、その条件分岐命令のモードを初期モード保持手段に、その条件分岐命令の種類を投機実行種類保持手段に出力し、投機実行状態指示手段のフラグをセットする。分岐判定手段により分岐する否かが判定されると、実行結果管理手段は、その判定結果に基づいて、初期モード保持手段、投機実行種類保持手段を参照して一時記憶手段の実行結果の有効無効を識別し、有効な実行結果を本来の格納先に転送し、無効な実行結果をクリアする。

【００２４】

【実施例】図１は本発明のプロセッサの構成図である。図１において、１はメモリであり、プログラムを構成する命令列やオペランドデータを記憶する。２は命令フェッチ部であり、メモリ１から命令をプリフェッチし、命令フェッチバッファＡ３または命令フェッチバッファＢ４に書き込む。この命令フェッチ部２は、フェッチした命令が条件分岐命令であるかどうかを検出し、検出結果に応じて書き込み先の命令フェッチバッファを切り替える。

【００２５】この命令フェッチ部２によって検出される分岐命令について説明する。例えば、プログラム言語Ｃの場合、一般的には、分岐するかしないかが不確定なif-then-else系の場合と、分岐する確率が高いループの場合と、分岐しない確率が高い場合の分岐動作がある。switch文の場合には実装方法によるが一般的にif-then-elseと同様に考えてよい。そこで、本実施例では、あらかじめ分岐の確率に応じて、分岐の確率が高い場合に高速に動作するループ系の分岐命令と、分岐の確率が不明な場合に高速に動作するif-then-else系の分岐命令と、分岐しない確率が高い場合に高速に動作する分岐命令の三種類を機械語命令の中に設けている。ループ系の分岐命令は、ループ向きの分岐命令で分岐する確率が高い命令（以後、Bcc_lと省略）であり、if-then-else系の分岐命令は分岐するかしないかが不確定な分岐命令（以後、Bcc_iと省略）である。このほかに、分岐しない確率が高い命令（以後、Bcc_nと省略）がある。

【００２６】分岐命令が命令フェッチ部２により検出されると、分岐命令に後続する命令列（分岐しないときに実行される命令列）と分岐先の命令列（分岐したときに

実行される命令列）の両方を読み出し、分岐命令に後続する命令列をそれまでと同じ命令フェッチバッファに、分岐先の命令列をもう１つの命令フェッチバッファに格納する。この読み出し動作は、命令の解読や実行とは非同期に、命令フェッチバッファＡ３または命令フェッチバッファＢ４を常に一杯にする。

【００２７】命令フェッチバッファＡ３は、命令フェッチ部２により読み出された命令をＦＩＦＯ（First In First Out）メモリである。命令フェッチバッファＢ４は、３と同様、ＦＩＦＯメモリである。５はセレクタであり、２つの命令フェッチバッファから出力される命令を切り替える。切り替えの指示は命令解読部７によって行われる。

【００２８】６は命令解読バッファであり、命令解読の対象となる命令を記憶し、本実施例では、６命令分の記憶容量を持つ。記憶容量については、少なくとも演算ユニット８～１１と同数以上が望ましい。命令解読部７は、命令フェッチバッファから命令解読バッファ６への命令の転送制御、命令解読バッファの命令解読、及び演算ユニットへの命令発行を行う。

【００２９】８は演算ユニットＡであり、ロード命令とストア命令を実行し、このうちロード命令の実行は、説明の便宜上２サイクルかかるとしている。９は演算ユニットＢであり、整数演算を処理する。１０は演算ユニットＣであり、整数演算を処理する。１１は演算ユニットＤであり、浮動小数点演算を処理する。演算ユニットＡ８～Ｄ１１は本実施例では説明を簡単化するため１サイクル（ロード命令は２サイクル）で命令の実行が完了するとしている。また、ユニットのパイプライン段数も本発明の主旨とは関係無いのでここでは定義しない。

【００３０】１２は演算ユニット管理テーブルであり、演算ユニットごとに、使用中であることを示す情報（ビット）を記憶する。このビットは、各演算ユニットによって、演算ユニットの命令実行開始時にセットされ、演算ユニットの命令実行終了時にクリアされる。１３は実行順序管理バッファであり、投機的に実行された命令の演算結果と、その結果が本来格納されるべきレジスタの番号と、その命令のモードと記憶する。実行順序管理バッファ１３の記憶形式は、図７に示すようにモード、レジスタ番号、演算結果を記憶する領域を有している。

【００３１】１４は初期モード保持回路であり、命令解読部が処理した分岐命令に付加されているモードを初期モードとして保持する。１５は投機実行種類保持回路であり、現在実行している分岐命令の種類（Bcc-1であるかBcc-iであるか）を保持する。１６は投機実行状態指示回路であり、現在投機実行中の状態か否かを示すフラグを保持する。

【００３２】１７は実行順序管理回路であり、上記１４～１６から得られる情報に応じて、演算ユニット８～１１の演算結果をレジスタファイル１９又は実行順序管理

バッファ１３に格納する。具体的には、投機実行中でないときは、演算結果をレジスタファイルに書き込み、投機実行中のときは、投機実行中の命令について、モード、演算結果（オペランド）の書き込み先のレジスタ番号を対応づけて実行順序管理バッファ１３に格納する。また、投機実行が終了したときは、実行順序管理バッファに格納された演算結果及びレジスタ番号に基づいてレジスタファイルに演算結果を転送する。

【００３３】１８は命令実行整合性維持回路であり、ＰＳＲの確定結果から条件分岐命令の条件の成否を判定し、その結果によって、命令解読部７に命令実行のやり直しの指示、実行順序管理バッファ１３内部の無効化、実行順序管理回路１７の転送指示を行う。より具体的には、条件分岐命令の種類によって、次の場合に分けられる。

【００３４】（１）Bcc-1命令について投機実行が行われている場合には、分岐先の命令のみ実行されているので、確定の結果から分岐しないことが判明したときには先行して実行した命令の結果を実行順序管理バッファ内から消去し、再度分岐しない側の命令を実行するように命令解読部に指示する。

（２）Bcc-i命令について投機実行が行われている場合には、分岐先の命令と後続する命令が実行されているので、分岐方向が確定したときには、それに応じて、後続命令または分岐先命令の結果を実行順序管理バッファ内から消去する。

【００３５】（３）Bcc-n命令について投機実行が行われている場合には、分岐命令に後続する命令のみ実行されているので、確定の結果から分岐することが判明したときには先行して実行した命令の結果を実行順序管理バッファ内から消去し、再度分岐する側の命令を実行するように命令解読部に指示する。１９はレジスタファイルであり、３２個のレジスタ及びＰＳＲを有し、命令の実行結果が格納される。

【００３６】２０はスコアボードであり、レジスタファイル１９の各レジスタごとに２ビットのフラグを有する。１つは、投機実行でない実行中の命令が、対応するレジスタを使用（リード又はライト）していることを示す。もう１つは、投機実行で実行中の命令が、対応するレジスタを使用していることを示す。図２は、図１の命令フェッチ部２の詳細な構成を示すブロック図である。同図において、２１はプログラムカウンタであり、メモリ１から読み出すべき命令のアドレスを出力し、命令が読み出されるごとにその内容をインクリメントする。

【００３７】２２は分岐命令検出回路であり、メモリ１から読み出された命令から条件分岐命令を検出する。２３は演算回路であり、検出された条件分岐命令に基づいて分岐先アドレスを算出する。２４はセレクタであり、プログラムカウンタ２１と演算回路２３の出力を選択してメモリ１に出力する。

【００３８】２５はフェッチ制御部であり、メモリ１からの命令読み出し、および、命令フェッチバッファＡ３または命令フェッチバッファＢ４への命令書き込みを制御する。詳しく言うと、通常は、プログラムカウンタ２１がインクリメントしながらアドレスを順に出力するように制御し、一方の命令フェッチバッファに命令を格納する。分岐命令検出回路２２により条件分岐命令が検出されると、その条件分岐命令が有する分岐先アドレスの情報を演算回路２３に入力して分岐先アドレスを算出させ、分岐先の命令も読み出しうるように制御し、他方の命令フェッチバッファに命令を格納する。

【００３９】図３は、図１の命令解読部７の詳細な構成を示すブロック図である。同図において、３１は分岐命令検出回路であり、図１のセレクタ５の出力から分岐命令（Bcc_l命令及びBcc_i命令を検出する。３２は転送制御回路であり、命令フェッチバッファＡ３または命令フェッチバッファＢ４の命令を、セレクタ５を介して命令解読バッファ６に転送する。この転送は、分岐命令検出回路３１の検出結果によってその動作が異なる。

【００４０】第１に、分岐命令が検出されていない場合、転送制御回路３２は、命令フェッチバッファＡ３又は命令フェッチバッファＢ４のうちどちらか命令が入っている方の出力ををセレクタ５に選択させて命令解読バッファ６に命令を転送させる。第２に、Bcc_l系命令（ループ向きの分岐命令で、分岐の確率が高い）が検出された場合、現在命令を読み出している命令フェッチバッファから、他方の命令フェッチバッファに切り替えて、命令解読バッファに命令を転送する。命令解読バッファ６には分岐先の命令ばかりが転送されることになる。

【００４１】第３に、Bcc_i系命令（if-then-else系の分岐命令で、分岐の確率が不確定）が検出された場合、命令の読み込みを現在使用している命令フェッチバッファと他方の命令フェッチバッファから交互に転送する。命令フェッチバッファには後続する命令（分岐しない時に実行される命令）と分岐先の命令とが混合して（交互に）転送されることになる。

【００４２】第４に、Bcc_n系命令（分岐しない確率が高い）が検出された場合、現在命令を読み出している命令フェッチバッファから、引き続き命令解読バッファに命令を転送する。命令解読バッファ６には分岐しない側の命令ばかりが転送されることになる。３３はモード付加回路であり、転送制御回路３２による上記の転送に際して、分岐命令による命令の制御フローが判るように各命令にモードを付加して、命令解読バッファ６に格納する。

【００４３】このモード設定ルールは、

（１）分岐命令に後続する命令列のモードは、現在のモードに" +０１" を加算する。

（２）分岐先の命令列のモードは、現在のモードに" +１０" を加算する。

の２つである。

【００４４】モードと命令のフローの一例を示したモード説明図を図４に示す。同図において、実線の矢印（↓）は命令列を、丸印（○）は条件分岐命令を、破線の矢印（）はループの戻り先（Bcc_l命令の分岐先）を、" " 内の数字はモードを示す。また、丸付き数字（❶、❷、❸）はループの繰り返し回数である（図２では３回繰り返す）。

【００４５】例えば、現在の命令列のモードが" ００" とすると、分岐命令に後続する命令列のモードは" +０１" を加算して" ０１" 、分岐先の命令列のモードは" +１０" を加算して" １０" として設定される。また、ループ処理において、ループを繰り返すごとにモードが変化するのは、モードが相対的に設定されているからである。

【００４６】このモード設定ルールを使用すれば分岐命令までの命令列と後続する命令列と分岐先の命令列を容易に判別でき、分岐先命令列に新たな分岐が存在し、木のように枝別れする場合でも制御フローを判別することが可能である。３４は命令解読回路であり、デコーダ３４ａ～３４ｆを備え、図１の命令解読バッファ６から入力される６つの命令を同時に解読する。ＤＥＣの数は、ここでは命令解読バッファ６の段数と同数であり、演算ユニット８～１１と同数以上が望ましい。

【００４７】３５はＰＳＲ変更命令検出回路であり、デコーダ３４ａ～３４ｆの解読結果が入力され、それぞれの命令がＰＳＲ（Program Status Register）の内容を変更するかどうかを検出する。ＰＳＲ内容が確定すると条件分岐命令の条件の成否が決定するので、この検出結果は、命令実行整合性維持回路１８に通知される。３６はデータ依存関係検出回路であり、デコーダ３４ａ～３４ｆから入力される解読結果に基づいて、１つの命令の実行結果を他の命令が利用するというデータ依存関係あるかどうかを検出する。具体的には、データ依存関係検出回路３６は、解読された６つの命令について、解読結果のレジスタフィールドを比較して、実行アドレスが前の命令のディスティネーション・レジスタが後の命令のソース・レジスタになっている場合、データ依存関係があると判定する。

【００４８】データ依存関係を調べる範囲については、命令解読バッファ６の命令と、命令解読バッファ６にまだ取り込まれていない命令との間のデータ依存関係は、調べる必要がない。というのは、前者と後者の命令は同時に実行されることがないからである。命令解読バッファ６内にある命令についてのみデータ依存関係を調べればよい。同時に実行されうるからである。ただし、命令解読バッファ６内の命令であっても、表１に示すように、分岐先の命令列における命令（Ｂ）と、分岐命令に後続する命令列における命令（ｃ）との間のデータ依存

15　　16

関係は、チェックしない。なぜなら、（B）と（C）は排他的な関係にあるからである。表１に本発明の実施例における命令列のデータ依存関係のチェックの有無を示す

【００４９】

【表１】

	(A)	(B)	(C)
分岐命令までの命令列（A）の命令	チェックする	チェックする	チェックする
後続する命令列（分岐しない側）の命令（B）		チェックする	チェックしない
分岐先命令列（C）の命令			チェックする

【００５０】３７はスコアボード管理回路であり、スコアボード２０を参照して命令の実行で使用されているレジスタの状況を判断し、また、デコーダ３４ａ～３４ｆの解読結果を見て、レジスタの内容を変更する命令があれば、そのレジスタに対応するスコアボード２０のフラグをセットする。３８は空き状態検知回路であり、演算ユニット管理テーブル１２を参照してどの演算ユニットが空いている（使用中でない）かを判断する。。

【００５１】３９は命令発行回路であり、セレクタ３９ａ～３９ｄを有し、命令発行制御回路４０の指示に従って、命令解読回路３４によって解読された命令のうち発行可能な命令を選択して演算ユニット８～１１に並列に発行する。これと同時に、命令発行回路３９は、発行する命令ごとに、投機実行であるかどうかを示す識別子を演算ユニットに格納する。この識別子は、本実施例では、”０”で投機実行でないことを、”１”で投機実行中であることを示す。

【００５２】命令発行制御回路４０は、上記３６～３８から得られる情報に基づいて、命令解読回路３４によって解読された命令のうち発行可能な命令を判別して、命令発行回路３９に発行の指示を出す。発行可能な命令かどうかは、命令ごとに、次の条件をチェックして全部満たすものを発行可能と判別する。

①６つの命令間において、前の命令とデータ依存関係がないこと。（データ依存関係検出回路３６の検出結果に基づいて、チェックする。）

②その命令のオペランドで指定されているレジスタが演算ユニットで実行中の命令に使用されていないこと。（スコアボード管理回路３７からの情報に基づいて、チェックする。）

③その命令を実行しうる演算ユニットが空いていること。（空き状態検知回路３８の検知結果に基づいて、チェックする。）

また、命令の発行と同時に、命令発行制御回路４０は、命令解読バッファ６から発行した命令を削除し、演算ユニット管理テーブル１２の対応するビットをセット（使用中）する。

【００５３】発行可能な命令が条件分岐命令である場合には、演算ユニットには命令発行せず、初期モード保持回路１４に分岐命令のモードを、投機実行種類保持回路１５にBcc-i命令であるかBcc-l命令であるかBcc-nであるかを、投機実行状態指示回路１６に投機実行中であることを示すフラグを、命令実行整合性維持回路１８にその命令の条件をセットする。この場合、これ以降は投機実行状態になる。

【００５４】以上のように構成された本発明の実施例における分岐処理装置について、最初にBcc-i命令を含む場合について、その動作を図５～図９及び表２を併用して説明する。図５は、Bcc-i命令を高速に処理する本実施例の動作を説明するための命令フローの一例である。図５において、矢印は命令実行順序を示し、命令Ｎ－３と命令Ｎ－２の間の矢印はデータの依存関係があることを、（）内のＬＤはロード命令、ＩＮＴは整数演算命令、ＦＰＵは浮動小数点演算命令を示す。命令Ｎ－２は、分岐命令の基になるＰＳＲを変更する命令である。

【００５５】まず、命令フェッチ部２の動作について説明しておく。命令フェッチ部２は、命令解読以下の処理とは非同期に動作する。命令フェッチ部２は、プログラムカウンタ２１を用いてメモリ１から命令を読み込み、命令フェッチバッファＡ３に格納する。メモリ１からの読み出し時に、分岐命令検出回路２２が分岐命令を検出すると、その分岐命令に後続する命令を命令フェッチバッファＡ３に格納するとともに、演算回路２３が分岐先アドレスを計算し、分岐先の命令をもう一つの命令フェッチバッファＢ４に格納する。

【００５６】この命令フェッチ部２の動作は、分岐命令がない場合にはどちらか一方の命令フェッチバッファが一杯になるまで、分岐命令がある場合には両方の命令フェッチバッファが一杯になるまで続けられる。命令フェッチバッファＡ３と命令フェッチバッファＢ４は同等で

あり、どちらを先に使用してもかまわない。これ以降の説明では命令フェッチバッファが常に一杯になっているとして説明を進める。

【００５７】図５において、命令Ｎ－３より以前に実行された命令については、本発明の説明とは無関係なので説明を加えない。また、命令フェッチ部２によって、図３の命令列のうち分岐命令までの命令列と分岐命令に後続する命令列は順次命令フェッチバッファＡ３に格納され、分岐先の命令列は命令フェッチバッファＢ４に格納されている。以下、順を追って説明する。

【００５８】（１）命令解読部７は、命令フェッチバッファＡ３から命令を読みだし、その命令にモードを付加して命令解読バッファ６に格納する。命令フェッチバッファＡ３に格納されている図６のＮ－３からＮ＋２までの命令は、命令解読部７によって読み出されてモード"００"が付加され、命令解読バッファ６に格納される。この時点の命令解読バッファ６の保持内容を図６に示す。

【００５９】（２）命令解読部７において、データ依存

関係検出回路３６は命令間のデータ依存関係をチェックし、スコアボード管理回路３７は使用中のレジスタをチェックし、空き状態検知回路３８は演算ユニット管理テーブル１２で演算ユニットの空き状況を判断する。これらの結果に応じて、命令発行制御回路４０は、発行可能な命令を判別し、命令発行回路３９に命令を投機実行であるか否か示す識別子を付けて発行させるとともに、当該命令を命令解読バッファ６から削除する。

【００６０】図６において、命令Ｎ－２は、命令Ｎ－３とデータ依存関係があるので最初は発行されない。Ｎ－３、Ｎ－１、Ｎ、Ｎ＋１の４つが発行される。投機実行ではないので各命令の識別子は"０"である。これらの命令発行後の演算ユニットの状態を表２の１行目に示す。各演算ユニットは、命令が投入されると、演算ユニット管理テーブル１２の対応するビットをセットする。表２は図５の命令フローを実行したときの演算ユニットでの命令実行の状況を示す。

【００６１】

【表２】

演算ユニットＡ	演算ユニットＢ	演算ユニットＣ	演算ユニットＤ
Ｎ－３	Ｎ－１	Ｎ	Ｎ＋１
Ｎ－３	Ｎ＋３	Ｍ	Ｎ＋２
	Ｎ－２	Ｎ＋４	Ｍ＋１
	Ｎ＋５	Ｍ＋２	
	Ｍ＋３	Ｍ＋４	

【００６２】（３）演算ユニットは、命令を実行した後、演算結果を実行順序管理回路１７に出力する。投機実行中の命令ではない（識別子が０である）ので、実行順序管理回路は、演算結果をレジスタファイル１９に直接書き込む。通常、分岐命令の無い命令列は（１）～（３）のように実行される。

（４）命令解読バッファ６は、図６において命令Ｎ－２と命令Ｎ＋２が残った状態になっている。命令解読部７の転送制御回路３２は、新たな命令を命令フェッチバッファＡ３から順に読み出し、命令解読バッファ６に格納する。その際、分岐命令検出回路３１が読み出した命令がBcc_i命令であることを検出すると、転送制御回路３２は、セレクタ５を交互に切り替えることによって、命令フェッチバッファＡ３から分岐命令に後続する命令（分岐しない場合に続いて実行する命令）と、命令フェッチバッファＢ４から分岐先命令を交互に読み出し、命令解読バッファに格納する。これと同時に、モード付加回路３３は、分岐先命令か否かを示すために、現状のモードを"００"とすると、後続する命令には"＋０１"を加算して"０１"、分岐先の命令には"＋１０"を加算して"１０"として設定する。すると命令解読バッファは図７のように命令が格納されることになる。

【００６３】（５）命令解読部７は（２）で述べたのと同様に、発行可能な命令を判別する。前のサイクルで発行された命令のうち、演算ユニットＡ８に発行された命令Ｎ－３は２サイクルかかるロード命令のためまだ実行が完了していない。そのため、この命令とデータ依存関係をもつ命令Ｎ－２は、もう１サイクル命令発行が遅らされる。この処理サイクルでは、命令Ｎ＋２、Bcc_i、命令Ｎ＋３、命令Ｍが発行可能であると判別される。命

令発行制御回路４０は、命令発行回路３９を介して、命令Ｎ＋２を演算ユニットＤ１１に、Bcc_i命令を命令実行整合性維持回路１８に、命令Ｎ＋３を演算ユニットＢ９に、命令Ｍを演算ユニットＣ１０に、識別子を付加して発行するとともに、命令解読バッファ６から削除する。これらの命令に付加される識別子は、同順に０、０、１、１である。命令発行後の各演算ユニットの状態を表２の２行目に示す。発行された各命令は、次の（６－１）～（６－４）のように実行される。

【００６４】（６－１）演算ユニットＤ１１は、命令Ｎ＋２が発行されると、演算ユニット管理テーブル１２の対応するビットをセットし、命令を実行する。演算ユニットＤ１１にて命令の実行が終了すると、実行結果が実行順序管理回路１７に出力される。これと同時に、演算ユニット管理テーブル１２の対応するビットがクリアされる。

【００６５】実行順序管理回路１７は、命令Ｎ＋２が投機実行モード下の実行ではない（識別子が０である）ので、演算結果を実行順序管理バッファ１３へは格納せず、レジスタファイル１９へ書き込む。

（６－２）Bcc-i命令は、演算ユニットには発行されずに、命令発行制御回路４０によって、初期モード保持回路１４にこの分岐命令自身のモード”００”が、投機実行種類保持回路１５にBcc-i命令であることが、投機実行状態指示回路１６に投機実行中であることを示すフラグが、命令実行整合性維持回路１８にその命令の条件がセットされる。これ以降は投機実行状態に入る。

【００６６】（６－３）演算ユニットＢ９は、命令Ｎ＋３が発行されると、演算ユニット管理テーブル１２の対応するビットをセットし、命令を実行する。演算ユニットＢ９にて命令の実行が終了すると、実行結果が実行順序管理回路１７に出力される。これと同時に、演算ユニット管理テーブル１２の対応するビットがクリアされる。

【００６７】実行順序管理回路１７は、この命令が投機実行下の命令（識別子が１）なので、実行順序管理バッファ１３に書き込み先のレジスタ番号とモード”０１”を書き込む。

（６－４）演算ユニットＣ１０は、命令Ｍが発行されると、演算ユニット管理テーブル１２の対応するビットをセットし、命令を実行する。演算ユニットＢ９にて命令の実行が終了すると、実行結果が実行順序管理回路１７に出力される。これと同時に、演算ユニット管理テーブル１２の対応するビットがクリアされる。

【００６８】実行順序管理回路１７は、この命令が投機実行下の命令（識別子が１）なので、実行順序管理バッファ１３に書き込み先のレジスタ番号とモード”１０”を書き込む。

（７）この時点で実行順序管理バッファ１３には図９の２行目まで積まれることになる。

【００６９】（８）また、命令解読バッファは図８のようになる。次のサイクルでは命令Ｎ－３が完了しているので、命令解読部７は一連の処理を行ない命令Ｎ－２、Ｎ＋４、Ｍ＋１を発行する。命令発行後の各演算ユニットの状態を表２の３行目に示す。これらの命令実行が終了した時点で、実行順序管理バッファ１３は図９の４行目まで積まれることになる。

【００７０】（９）上記のように演算ユニットは投入された命令を実行し、新たな命令を処理可能になれば演算ユニット管理テーブル１２の対応するビットをクリアする。命令解読部７は、新たな命令を処理できるようになれば、命令の発行処理を再開する。

（１０）PSRを変更する命令Ｎ－２は、表２の３行目に示すように演算ユニットＢで実行される。命令Ｎ－２が完了しPSRが確定すると、演算ユニットＢは投機実行状態指示回路１６のフラグをクリアする。命令Ｎ－２が完了しても、投機実行は、表２の４行目まで実行され、実行順序管理バッファ１３には図９の６行目まで投機実行中の命令が登録される。

【００７１】（１１）命令実行整合性維持回路１８は、投機実行状態指示回路１６のフラグがクリアされたことから、投機実行種類保持回路１５よりBcc_i命令であることと、PSRの値に基づいて、分岐するか否かを判断する。この例では分岐すると仮定する。命令実行整合性維持回路１８は、Bcc_i命令であることから実行順序管理バッファ１３内部の後続する命令列（分岐しない側の命令列）の演算結果はもう必要がないので、初期モード保持回路１４の初期モード”００”に”＋０１”を加算して、モードが”０１”である命令の演算結果を無効化する。図９において命令Ｎ＋３，Ｎ＋４，Ｎ＋５は無効化（クリア）される。また、命令解読部７に対して分岐することを通知し、以降分岐先の命令のみを命令解読バッファ６に取り込むようにさせる。

【００７２】（１２）実行順序管理回路１７は、命令実行整合性維持回路１８より、分岐が確定し、投機実行モードが終了したことを通知される。そして、実行順序管理バッファ１３内に存在する分岐先の命令列（モードが”１０”の命令）に対しては、演算結果を対応するレジスタファイル１９のレジスタに格納する。

以上のように、条件分岐命令Bcc_iの場合の投機実行でなされた処理は、分岐することが確定した後の処理と、連続性を維持している。しかも、およそ半分の投機実行結果は有効になるから、その分高速に処理されることになる。

【００７３】続いて、Bcc-1命令を含む場合について、その動作を図１０～図１４及び表３を併用して説明する。図１０はBcc_1命令高速に処理する本実施例の動作を説明するための命令フローの一例である。命令実行順序は矢印の通りである。また、命令Ｎ＋４と命令Ｎ＋７とはデータの依存関係があるとする。命令Ｎ＋７は条件

分岐命令の分岐判定の基になるPSRを変更する命令である。同図において、命令Ｎまでに実行されている命令があるが本発明の説明には関係しないので説明を加えない。また、命令フェッチ部２によって、図１０の命令列のうち分岐命令までの命令列と後続する命令列は順次命令フェッチバッファＡ３に格納されており、分岐先命令列は命令フェッチバッファＢ４に格納されている。以後、順を追って説明する。

【００７４】（１）命令解読部７は命令フェッチバッファＡ３より命令を読みだし、その命令にモードを付加して命令解読バッファ６に格納する。格納される命令のモードは"０１"とする。今、命令解読バッファ６は図１１の状態になっているとする。

（２）命令解読部７において、データ依存関係検出回路３６は命令間のデータ依存関係をチェックし、スコアボード管理回路３７は使用中のレジスタをチェックし、空

き状態検知回路３８は演算ユニット管理テーブル１２で演算ユニットの空き状況を判断する。これらの結果に応じて、命令発行制御回路４０は、発行可能な命令を判別し、命令発行回路３９に命令を投機実行であるか否か示す識別子を付けて発行させるとともに、当該命令を命令解読バッファ６から削除する。

【００７５】図１１において、命令Ｎ＋７は、命令Ｎ＋４とデータ依存関係があるので最初は発行されない。Ｎ＋４，Ｎ＋５，Ｎ＋６，Ｎ＋８の４つが発行される。投機実行ではないので各命令の識別子は"０"である。これらの命令発行後の演算ユニットの状態を表３の１行目に示す。表３は同実施例における図８の命令フローを実行したときの演算ユニットでの命令実行の状況を示す。

【００７６】

【表３】

演算ユニットA	演算ユニットB	演算ユニットC	演算ユニットD
N+4	N+5	N+6	N+8
N+4	N	N+1	N+3
	N+7	N+2	
N+4	N+5	N+6	N+8
N+4	N	N+1	N+3

【００７７】各演算ユニットは、命令が投入されると、演算ユニット管理テーブル１２の対応するビットをセットする。演算ユニットは、命令を実行した後、演算結果を実行順序管理回路１７に出力する。投機実行中の命令ではない（識別子が０である）ので、実行順序管理回路１７は、演算結果をレジスタファイル１９に直接書き込む。

【００７８】（３）命令解読バッファ６は、図１１において命令Ｎ＋７と命令Bcc_1が残った状態になっている。命令解読部７の転送制御回路３２は、新たな命令列を命令フェッチバッファＡ３から読みだす。読みだした命令がBcc_1命令であることを検出すると、セレクタ５を切り替えて命令フェッチバッファＢ４から分岐先の命令のみを読み出し、命令解読バッファ６に格納する。これと同時に、モード付加回路３３は、分岐先の命令か否かを示すために、現状のモードを"０１"とすると、分岐先の命令には"＋１０"を加算して"１１"として設定する。すると命令解読バッファは図１２のように格納されたことになる。

【００７９】（４）命令解読部７は（２）で述べたのと同様に、発行可能な命令を判別する。前のサイクルで発行された命令のうち、演算ユニットＡ８に発行された命令Ｎ＋４は２サイクルかかるロード命令であり、まだ実行が完了しない。そのため、この命令とデータ依存関係を持つ命令Ｎ＋７は、もう１サイクル命令発行が遅らされる。この処理サイクルでは、Bcc_1、命令Ｎ、命令Ｎ＋１、命令Ｎ＋３が発行可能であると判別される。命令発行制御回路４０は、命令発行回路３９を介して、Bcc_1命令を命令実行整合性維持回路１８に、命令Ｎを演算ユニットＢ９に、命令Ｎ＋１を演算ユニットＣ１０に、命令Ｎ＋３を演算ユニットＤ１１に、識別子を付加して発行するとともに、命令解読バッファ６から削除する。これらの命令に付加される識別子は、同順に０、１、１、１である。

【００８０】命令発行後の各演算ユニットの状態を表３の２行目に示す。発行された各命令は、次の（５－１）

～（5－4）のように実行される。

（5－1）Bcc-1命令は、演算ユニットには発行されずに、命令発行制御回路40によって、初期モード保持回路14にこの分岐命令自身のモード"01"が、投機実行種類保持回路15にBcc-1命令であることが、投機実行状態指示回路16に投機実行中であることを示すフラグが、命令実行整合性維持回路18にその命令の条件がセットされる。これ以降は投機実行状態に入る。

【0081】（5－2）演算ユニットB9は、命令Nが発行されると、演算ユニット管理テーブル12の対応するビットをセットし、命令を実行する。演算ユニットB9にて命令の実行が終了すると、実行結果が実行順序管理回路17に出力される。これと同時に、演算ユニット管理テーブル12の対応するビットがクリアされる。実行順序管理回路17は、この命令が投機実行下の命令（識別子が1）なので、実行順序管理バッファ13に書き込み先のレジスタ番号とモード"11"を書き込む。

【0082】（5－3）演算ユニットC10は、命令N＋1が発行されると、演算ユニット管理テーブル12の対応するビットをセットし、命令を実行する。演算ユニットB9にて命令の実行が終了すると、実行結果が実行順序管理回路17に出力される。これと同時に、演算ユニット管理テーブル12の対応するビットがクリアされる。

【0083】実行順序管理回路17は、この命令が投機実行下の命令（識別子が1）なので、実行順序管理バッファ13に書き込み先のレジスタ番号とモード"11"を書き込む。

（5－4）演算ユニットD11は、命令N＋3が発行されると、演算ユニット管理テーブル12の対応するビットをセットし、命令を実行する。演算ユニットD11にて命令の実行が終了すると、実行結果が実行順序管理回路17に出力される。これと同時に、演算ユニット管理テーブル12の対応するビットがクリアされる。

【0084】実行順序管理回路17は、命令N＋2が投機実行下の実行である（識別子が1である）ので、演算結果をレジスタファイル19には書き込まず、実行順序管理バッファ13へは格納する。

（6）この時点で、実行順序管理バッファ13には図14の3行目まで積まれることになる。

【0085】（7）また、命令解読バッファ6は図13のようになる。次のサイクルでは命令N＋4が完了しているので、命令解読部7は一連の処理を行ない命令N＋7，N＋2を発行する。命令発行後の演算ユニットの状態を表3の3行目に示す。これらの命令実行が終了した時点で、実行順序管理バッファには図14の4行目まで積まれることになる。

【0086】（8）上記のように演算ユニットは投入された命令を実行し、新たな命令を処理可能になれば演算ユニット管理テーブル12の対応するビットをクリアする。命令解読部7は、新たな命令を処理できるようになれば、命令の発行処理を再開する。

（9）PSRを変更する命令N＋7は、表3の3行目に示すように演算ユニットBで実行される。命令N＋7が完了しPSRが確定すると、演算ユニットBは投機実行状態指示回路16のフラグをクリアする。命令N＋7が完了しても、投機実行は、表3の4行目まで実行される。

【0087】（10）命令実行整合性維持回路18は、投機実行状態指示回路16がクリアされたことから、投機実行種類保持回路15よりBcc_1命令であること、PSRの値に基づいて、分岐するか否かを判断する。ここでは分岐すると仮定する。

（11）実行順序管理回路17は、命令実行整合性維持回路18より、分岐が確定し、投機実行モードが終了したことを通知される。そして、実行順序管理バッファ13内に存在する分岐先の命令列（モードが"11"の命令）に対しては、演算結果を対応するレジスタファイル19のレジスタに格納する。

【0088】以上のように、条件分岐命令Bcc_1の投機実行および分岐確定後の処理が、投機実行しない場合と同様に整合性を維持して実行できる。また、（10）にて分岐しないと確定した場合には以下のようになる。

（11）命令実行整合性維持回路は、分岐しないと判断した後、投機実行種類保持回路15を参照してBcc_1命令であることから、実行順序管理バッファ13内の分岐先命令の演算結果をすべて無効化する。図14において命令N，N＋1，N＋3，N＋2は無効化（クリア）される。また、命令解読部7に対して分岐しないことを通知し、再度命令フェッチバッファAに残されている後続する命令を命令解読バッファ6に取り込むよう処理をやり直しさせる。

【0089】（12）実行順序管理回路17は、命令実行整合性維持回路18より、分岐しないことが確定し、投機実行モードが終了したことを通知される。そして、実行順序管理バッファ内の命令の演算結果はすべて無効化されているため何もしない。

以上のように、条件分岐命令Bcc_1による投機実行でなされた処理は、分岐しないことが確定した後の処理に生かされることはなく、高速化には寄与していない。しかし、条件分岐命令Bcc_1は分岐しない確率が低いので、このようなケースが起こる確率も低く、1回の投機実行のみを見るのではなく、プログラム全体の全ての投機実行を併せて見ると、高速化されることになる。また、このケースでも、条件分岐命令より前の処理は、分岐しないことが確定した後の処理と連続性を維持している。

【0090】最後に、Bcc-n命令を含む場合について、その動作を図15～図19及び表4を併用して説明する。図15はBcc-n命令高速に処理する本実施例の動作を説明するための命令フローの一例である。先のBcc-1命令の説明で述べた（1）（2）までは、Bcc_n命令の

場合も同じであるので、ここでは、説明を省略する。

【００９１】（３）命令解読バッファ６は、図１６において命令Ｎ＋７と命令Bcc-nが残った状態になっている。命令解読部７の転送制御回路３２は、新たな命令列を命令フェッチバッファＡ３から読みだす。読みだした命令がBcc-n命令であることを検出すると、引き続き命令フェッチバッファＡ３から分岐命令に後続する（分岐しない側の）命令のみを読み出し、命令解読バッファ６に格納する。これと同時に、モード付加回路３３は、分岐先の命令か否かを示すために、現状のモードを"０１"とすると、分岐先の命令には"＋０１"を加算して"１０"として設定する。すると命令解読バッファは図１７のように格納されることになる。

【００９２】（４）命令解読部７は、発行可能な命令を判別する。前のサイクルで発行された命令のうち、演算ユニットＡ８に発行された命令Ｎ＋４は２サイクルかかるロード命令であり、まだ実行が完了しない。そのため、この命令とデータ依存関係を持つ命令Ｎ＋７は、もう１サイクル命令発行が遅らされる。この処理サイクルでは、Bcc-n、命令Ｎ＋９、命令Ｎ＋１０、命令Ｎ＋１１が発行可能であると判別される。命令発行制御回路４０は、命令発行回路３９を介して、Bcc-n命令を命令実行整合性維持回路１８に、命令Ｎ＋９を演算ユニットＢ９に、命令Ｎ＋１０を演算ユニットＣ１０に、命令Ｎ＋１１を演算ユニットＤ１１に、識別子を付加して発行するとともに、命令解読バッファ６から削除する。これらの命令に付加される識別子は、同順に０、１、１、１である。

【００９３】命令発行後の各演算ユニットの状態を表４の２行目に示す。表４は同実施例における図１５の命令フローを実行したときの演算ユニットでの命令実行の状況である。

【００９４】

【表４】

演算ユニットA	演算ユニットB	演算ユニットC	演算ユニットD
N+4	N+5	N+6	N+8
N+4	N+9	N+10	N+11
N+13	N+7	N+12	
N+13	N+14	N+15	
	N+16	N+17	

【００９５】発行された各命令は、次の（５－１）～（５－４）のように実行される。

（５－１）Bcc-n命令は、演算ユニットには発行されずに、命令発行制御回路４０によって、初期モード保持回路１４にこの分岐命令自身のモード"０１"が、投機実行種類保持回路１５にBcc-n命令であることが、投機実行状態指示回路１６に投機実行中であることを示すフラグが、命令実行整合性維持回路１８にその命令の条件がセットされる。これ以降は投機実行状態に入る。

【００９６】（５－２）演算ユニットＢ９は、命令Ｎ＋９が発行されると、演算ユニット管理テーブル１２の対応するビットをセットし、命令を実行する。演算ユニットＢ９にて命令の実行が終了すると、実行結果が実行順序管理回路１７に出力される。これと同時に、演算ユニット管理テーブル１２の対応するビットがクリアされる。

【００９７】実行順序管理回路１７は、この命令が投機実行下の命令（識別子が１）なので、実行順序管理バッファ１３に書き込み先のレジスタ番号とモード"１０"を書き込む。

（５－３）演算ユニットＣ１０は、命令Ｎ＋１０が発行されると、演算ユニット管理テーブル１２の対応するビットをセットし、命令を実行する。演算ユニットＢ９にて命令の実行が終了すると、実行結果が実行順序管理回路１７に出力される。これと同時に、演算ユニット管理テーブル１２の対応するビットがクリアされる。

【００９８】実行順序管理回路１７は、この命令が投機実行下の命令（識別子が１）なので、実行順序管理バッファ１３に書き込み先のレジスタ番号とモード"１０"を書き込む。

（５－４）演算ユニットＤ１１は、命令Ｎ＋１１が発行されると、演算ユニット管理テーブル１２の対応するビットをセットし、命令を実行する。演算ユニットＤ１１にて命令の実行が終了すると、実行結果が実行順序管理

回路１７に出力される。これと同時に、演算ユニット管理テーブル１２の対応するビットがクリアされる。

【００９９】実行順序管理回路１７は、命令Ｎ＋２が投機実行下の実行である（識別子が１である）ので、演算結果をレジスタファイル１９には書き込まず、実行順序管理バッファ１３へ格納する。

（６）この時点で、実行順序管理バッファ１３には図１９の３行目まで積まれることになる。

【０１００】（７）また、命令解読バッファ６は図１８のようになる。次のサイクルでは命令Ｎ＋４が完了しているので、命令解読部７は一連の処理を行ない命令Ｎ＋７、Ｎ＋１２、Ｎ＋１３を発行する。命令発行後の演算ユニットの状態を表４の３行目に示す。これらの命令実行が終了した時点で、実行順序管理バッファには図１９の５行目まで積まれることになる。

【０１０１】（８）上記のように演算ユニットは投入された命令を実行し、新たな命令を処理可能になれば演算ユニット管理テーブル１２の対応するビットをクリアする。命令解読部７は、新たな命令を処理できるようになれば、命令の発行処理を再開する。

（９）PSRを変更する命令Ｎ＋７は、表４の３行目に示すように演算ユニットＢで実行される。命令Ｎ＋７が完了しPSRが確定すると、演算ユニットＢは投機実行状態指示回路１６のフラグをクリアする。命令Ｎ＋７が完了しても、投機実行は、表４の４行目まで実行される。

【０１０２】（１０）命令実行整合性維持回路１８は、投機実行状態指示回路１６がクリアされたことから、投機実行種類保持回路１５よりBcc-n命令であること、PSRの値に基づいて、分岐するか否かを判断する。ここでは分岐しないと仮定する。

（１１）実行順序管理回路１７は、命令実行整合性維持回路１８より、分岐しないことが確定し、投機実行モードが終了したことを通知される。そして、実行順序管理バッファ１３内に存在する分岐命令に後続する命令列（モードが" １０" の命令）に対しては、演算結果を対応するレジスタファイル１９のレジスタに格納する。

【０１０３】以上のように、条件分岐命令Bcc_nによる投機実行でなされた処理は、分岐しないことが確定した後の処理と、連続性を維持している。しかも、全ての投機実行結果は有効であるから、最も効率よく高速に処理されることになる。また、（１０）にて分岐すると確定した場合には以下のようになる。

（１１）命令実行整合性維持回路は、分岐すると判断した後、投機実行種類保持回路１５を参照してBcc-n命令であることから、実行順序管理バッファ１３内の分岐先命令の演算結果をすべて無効化する。図１９において命令Ｎ＋９以降の命令は全て無効化（クリア）される。また、命令解読部７に対して分岐することを通知し、再度命令フェッチバッファＡに分岐先の命令を命令解読バッファ６に取り込むよう処理をやり直しさせる。

【０１０４】（１２）実行順序管理回路１７は、命令実行整合性維持回路１８より、分岐しないことが確定し、投機実行モードが終了したことを通知される。そして、実行順序管理バッファ内の命令の演算結果はすべて無効化されているため何もしない。

以上のように、条件分岐命令Bcc_nによる投機実行でなされた処理は、分岐することが確定した後の処理に生かされることはなく、高速化には寄与していない。しかし、条件分岐命令Bcc_nは分岐する確率が低いので、このようなケースが起こる確率も低く、１回の投機実行のみを見るのではなく、プログラム全体の全ての投機実行を併せて見ると、高速化されることになる。また、このケースでも、条件分岐命令より前の処理は、分岐することが確定した後の処理と連続性を維持している。

【０１０５】なお、本実施例では、演算ユニットのパイプラインは説明を簡単にするために１段にしている。複数段にしても演算ユニット管理テーブル１２などの変更だけで済むが、本発明の要旨には関係しない。本実施例では、命令解読バッファ６に積む順番は交互に積んだが、この順番は逆でも、また、命令は１つづつ交互にする必要はない。また、命令解読バッファ６自身、必ずしも備える必要はない。その場合、命令フェッチバッファＡ３、Ｂ４からセレクタ５を介して、直接命令解読回路３４の各デコーダに対して命令を転送すればよい。

【０１０６】本実施例では、演算ユニットＡ８～Ｄ１１は説明を簡単にするため、それぞれ機能を限定したがすべて同等の機能をもつユニットの構成でもよい。本実施例では、演算結果の格納先がレジスタファイル１９であるか、実行順序管理バッファ１３であるかを区別するために、命令発行の際に命令に識別子を付加することで実現したが、識別子の代わりにＰＣ値を使用してもよい。また、命令解読部７は、命令に識別子を付加せずに、実行順序管理回路１７に格納先を指示してもよい。

【０１０７】本実施例では、説明を簡単にするため演算ユニットの演算結果を最終的にレジスタファイル１９のレジスタに格納する命令（オペランドがレジスタにある命令）を用いているが、これに限らず、演算結果を最終的にメモリに格納する命令（オペランドがメモリにある命令）が用いられていてもよい。その場合例えば、実行順序管理バッファ１３は、演算結果、それが本来格納されるべきメモリのアドレスを指す情報、モードを一時的に記憶することになり、かつ、実行順序管理回路１７は、演算結果をメモリに書き込む機能を持っていればよい。

【０１０８】本実施例では投機実行状態保持回路では状態を１つしか持てないようにしているため、投機実行している分岐命令は１つに限られているだけで、複数投機実行してもよい。本実施例では命令解読部で命令が発生する毎に命令の制御フローが判るように、以下のルールでモードを付加した。例えば、現在の命令列のモード

が"００"とすると、後続する命令列のモードは"＋０１"を加算して"０１"、分岐先の命令のモードは"＋１０"を加算して"１０"として設定する。このルールを使用すれば分岐命令までの命令列と後続する命令列と分岐先の命令列を容易に判断できる。しかし、これは区別するための一例であり他の方法でも可能である。

【０１０９】本実施例では投機実行種類保持回路１５では状態を１つしか持てないようにしているため、投機実行の基になる分岐命令は１つに限られている。しかし、命令列を区別するモードの種類を増加し、初期モード保持回路１４、投機実行種類保持回路１５、投機実行状態指示回路１６、命令実行整合性維持回路１８を複数備えることにより、PSRを変更する命令と分岐命令の対応関係が明確になるので、複数の分岐命令に対して投機実行することが可能になる。

【０１１０】本実施例では、データ依存関係については解読部は命令発行をout-of-order発行で行なったが、in-orderでも本発明の主旨には影響しない。以上説明してきたように本発明の情報処理装置は、Bcc-l、Bcc-i、Bc c-nの３種類の分岐命令を使い分けることによって、分岐の確率に合わせて、より高速化が図れる。例えば、プログラミング言語Cで記述されたプログラムにおいて、コンパイラが、分岐の確率が一般に高いforループに対してはBcc_l命令を使用し、分岐の確率が不確定なif-then-else系の命令に対してはBcc_i命令を使用する。その上、ループ内の命令列をできるだけ同時実行可能なようにスケジューリングすればその効果はより大きくなる。また、if-then-else系の命令の場合、どちらかに予測してもはずれたときの代償が大きいと同時に、分岐をどちらかに予測して投機実行しても、データ依存関係により命令を１つずつしか実行できない場合も多くあり、このような場合並列実行率が上がらず、結局両方を実行した場合の方が効果が大きい。

【０１１１】

【発明の効果】本発明のプロセッサによれば、条件分岐命令の条件が確定していなくても、その条件分岐命令の種類に応じて、投機的に実行する命令列の命令を変えることにより、インターロックが少なく、分岐処理を含むプログラムを高速に実行することができる。特に、複数命令の同時実行する場合には分岐の条件を変更する命令と条件分岐命令間の時間差が少なくなり、この効果は大きい。

【０１１２】具体的には、つぎの３つ場合を使い分ける。

①第１の種類の条件付き分岐命令の場合には、投機的に分岐先の命令を複数の演算ユニットに発行し、その演算結果を一時的に格納しておくとともに、条件が確定し分岐しないことが判明した場合には、分岐しない側の命令列を実行することにより、分岐命令の先行処理が可能となり、分岐処理が高速化できるという効果がある。この第１種類の条件付き分岐命令は、分岐する確率が高い場合に用いられる。

【０１１３】②第２の種類の条件付き分岐命令の場合には、投機的に分岐先の命令と分岐しない側の命令を複数の演算ユニットに発行し、その演算結果を一時的に格納しておくとともに、条件が確定した場合には、確定した側の演算結果のみを有効とすることにより、分岐命令の先行処理が可能となり、分岐処理が高速化できるという効果がある。この第２の種類の条件付き分岐命令は、分岐する確率が不明な場合に用いられる。

【０１１４】③第３の種類の条件付き分岐命令の場合には、投機的に分岐しない側の命令を複数の演算ユニットに発行し、その演算結果を一時的に格納しておくとともに、条件が確定し分岐することが判明した場合には、分岐する側の命令列を実行することにより、分岐命令の先行処理が可能となり、分岐処理が高速化できるという効果がある。この第３の条件付き分岐命令は、分岐しない確率が高い場合に用いられる。

【図面の簡単な説明】

【図１】本発明の実施例におけるプロセッサの構成図、

【図２】同実施例における命令フェッチ部の詳細な構成を示すブロック図である。

【図３】命令解読部の詳細な構成を示すブロック図である。

【図４】図４は同実施例における命令のフローに付加されたモードを示したモード説明図である。

【図５】同実施例におけるBcc_i命令を含む命令フロー図である。

【図６】同実施例における図５の命令フローを処理するときの命令解読バッファの説明図（その１）である。

【図７】同上（その２）である。

【図８】同上（その３）である。

【図９】同実施例における図５の命令フローを処理するときの実行順序管理バッファの説明図、

【図１０】同実施例におけるBcc_l命令を含む命令フロー図である。

【図１１】同実施例における図１０の命令フローを処理するときの命令解読バッファ６の説明図（その１）である。

【図１２】同上（その２）である。

【図１３】同上（その３）である。

【図１４】同実施例における図１０の命令フローを処理するときの実行順序管理バッファの説明図、

【図１５】同実施例におけるBcc_n命令を含む命令フロー図である。

【図１６】同実施例における図１５の命令フローを処理するときの命令解読バッファ６の説明図（その１）である。

【図１７】同上（その２）である。

【図１８】同上（その３）である。

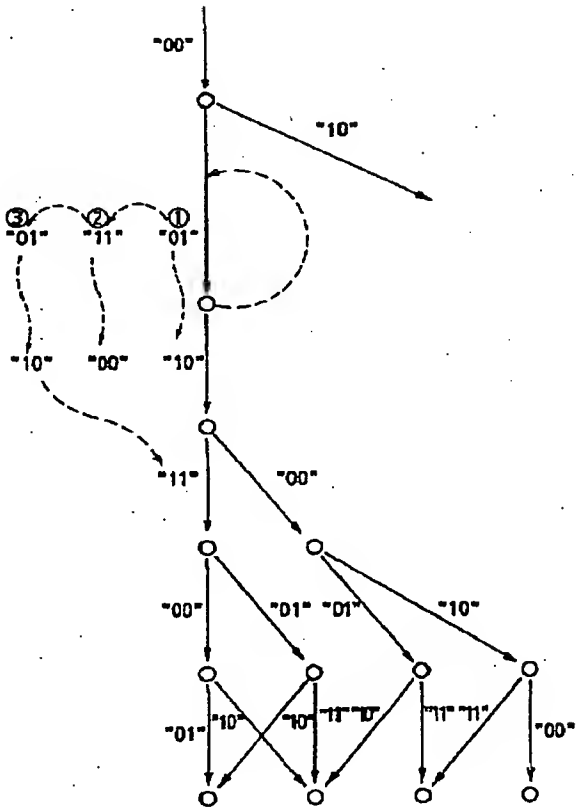
31　　32

【図１９】同実施例における図１５の命令フローを処理するときの実行順序管理バッファの説明図である。

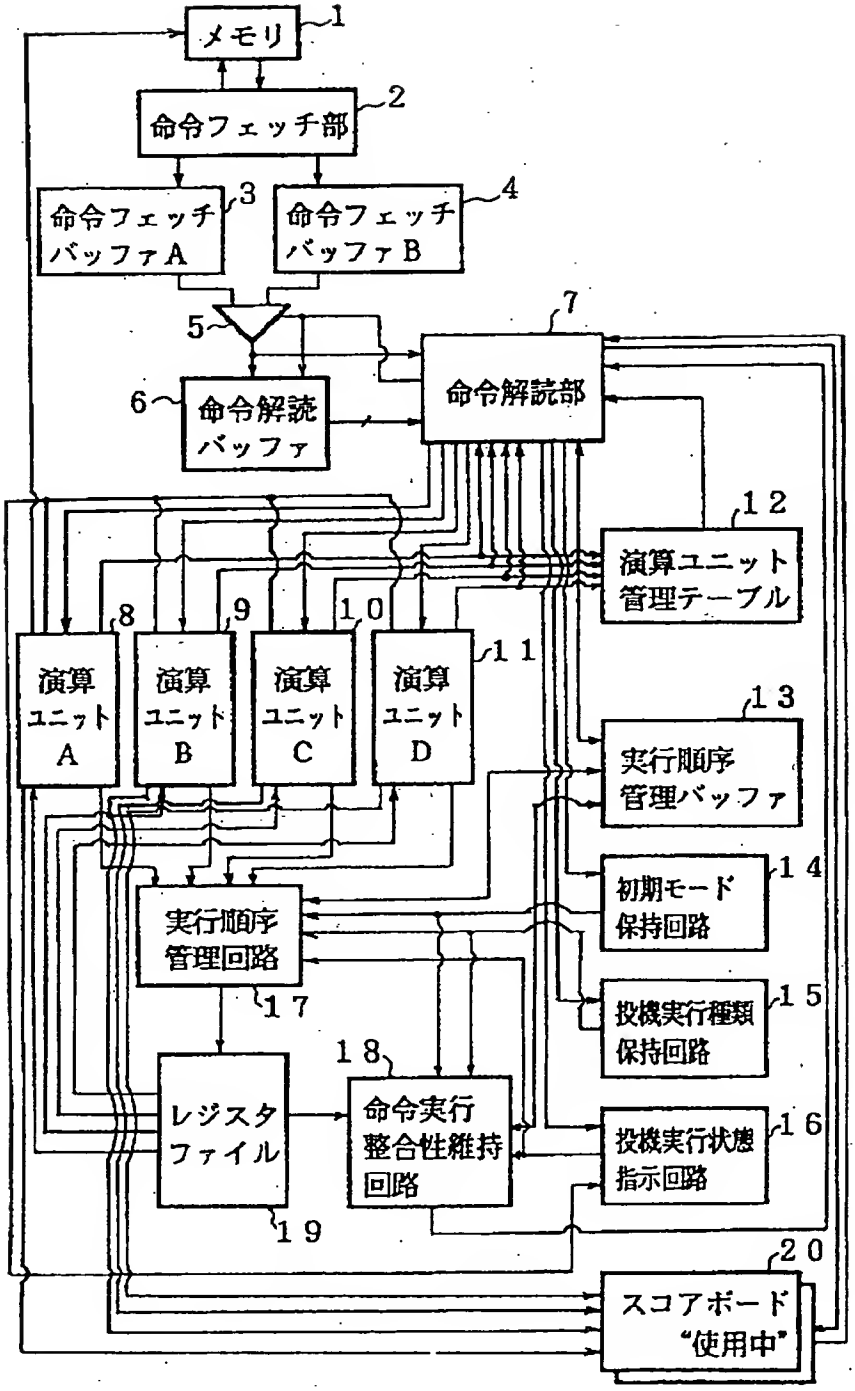
【符号の説明】

１　メモリ
２　命令フェッチ部
３　命令フェッチバッファＡ
４　命令フェッチバッファＢ
５　セレクタ
６　命令解読バッファ
７　命令解読部
８　演算ユニットＡ
９　演算ユニットＢ
１０　演算ユニットＣ
１１　演算ユニットＤ
１２　演算ユニット管理テーブル
１３　実行順序管理バッファ
１４　初期モード保持回路
１５　投機実行種類保持回路
１６　投機実行状態指示回路
１７　実行順序管理回路
１８　命令実行整合性維持回路
１９　レジスタファイル
２０　スコアボード
２１　プログラムカウンタ
２２　分岐命令検出回路
２３　演算回路
３１　分岐命令検出回路
３２　転送制御回路
３３　モード付加回路
３４　命令解読回路
３４ａ～３４ｆ　デコーダ
３６　データ依存関係検出回路
３７　スコアボード管理回路
３８　状態検知回路
３９　命令発行回路
３９ａ～３９ｄ　セレクタ
４０　命令発行制御回路

【図４】

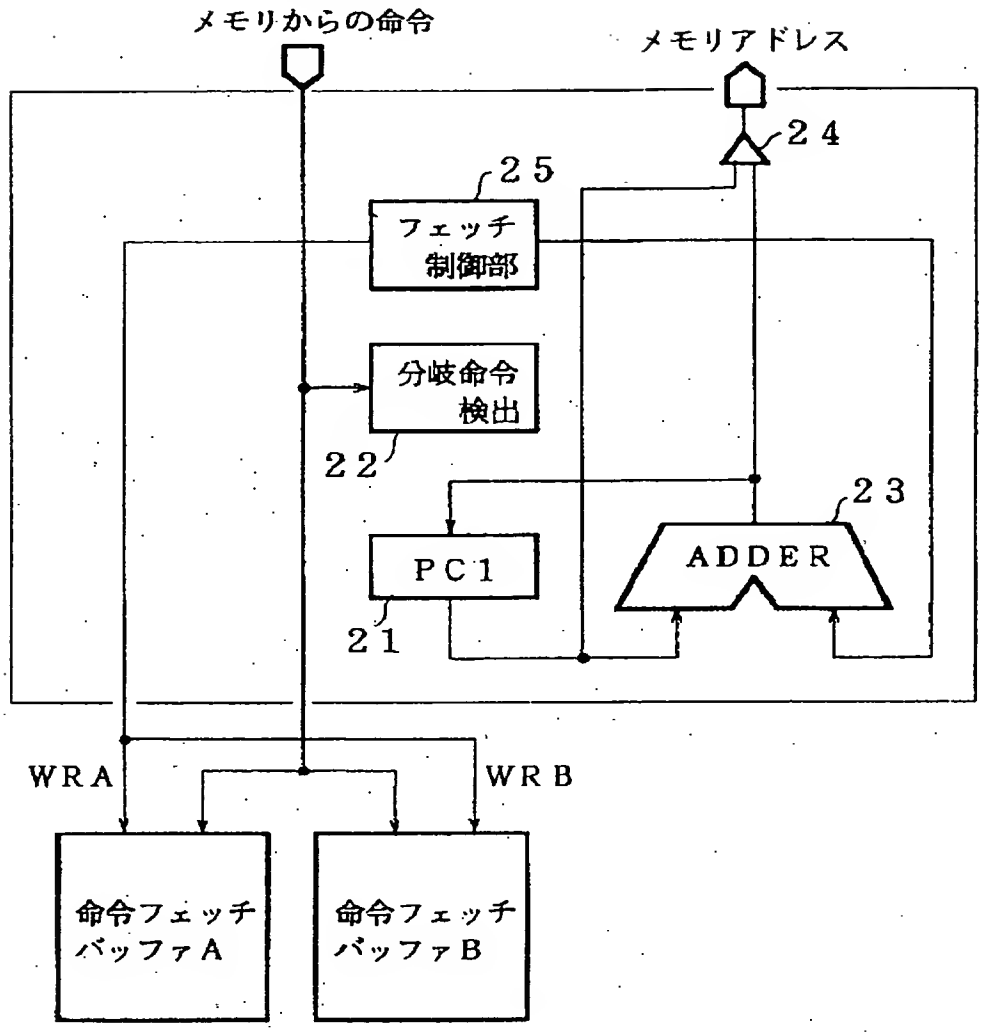


【図１】

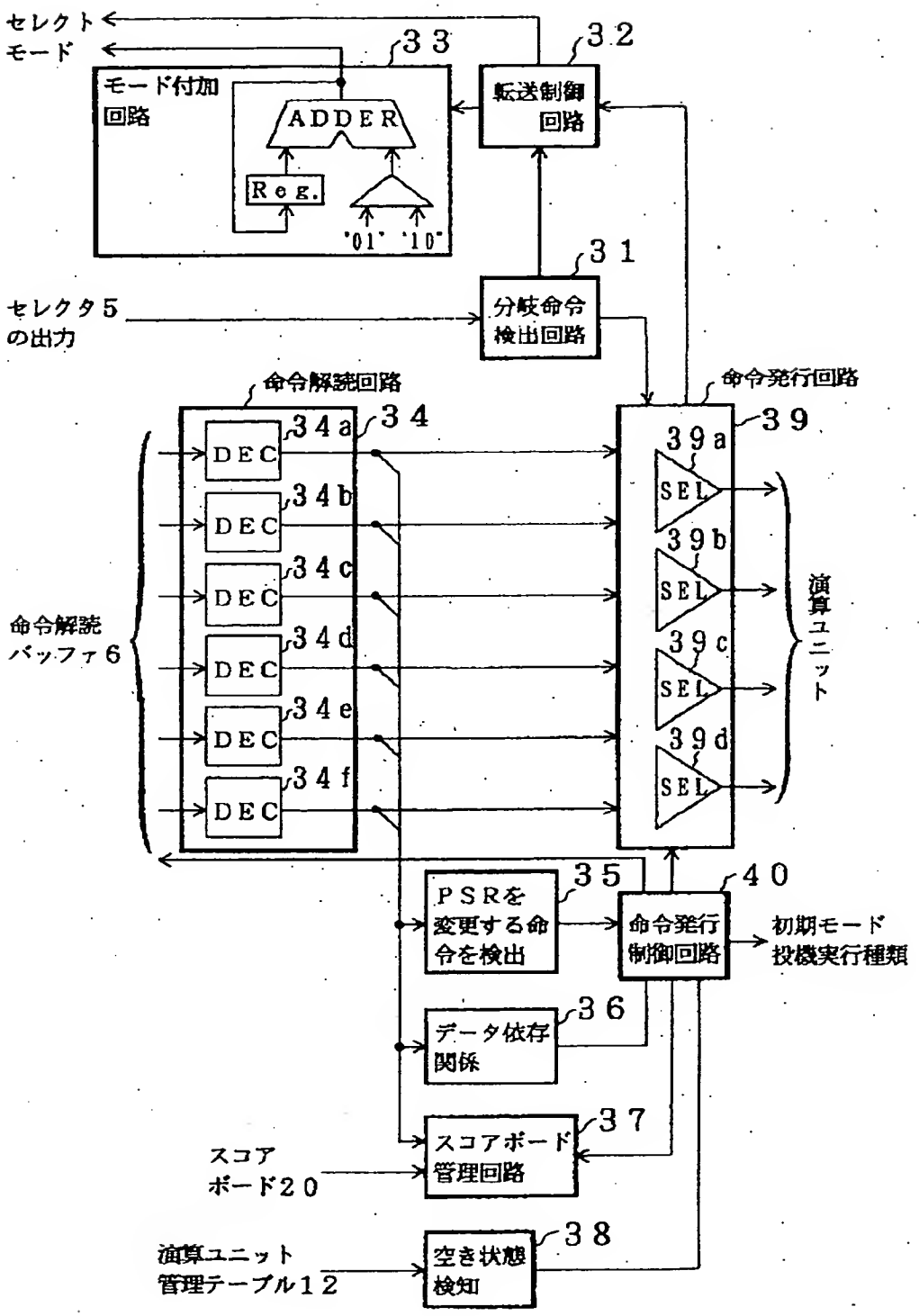

メモリ
1
命令フェッチ部
2
命令フェッチ
バッファA
3
命令フェッチ
バッファB
4
5
7
命令解読部
6
命令解読
バッファ
12
演算ユニット
管理テーブル
8
9
10
11
演算
ユニット
A
演算
ユニット
B
演算
ユニット
C
演算
ユニット
D
13
実行順序
管理バッファ
14
初期モード
保持回路
実行順序
管理回路
17
15
投機実行種類
保持回路
18
レジスタ
ファイル
命令実行
整合性維持
回路
16
投機実行状態
指示回路
19
20
スコアボード
“使用中”

【図２】

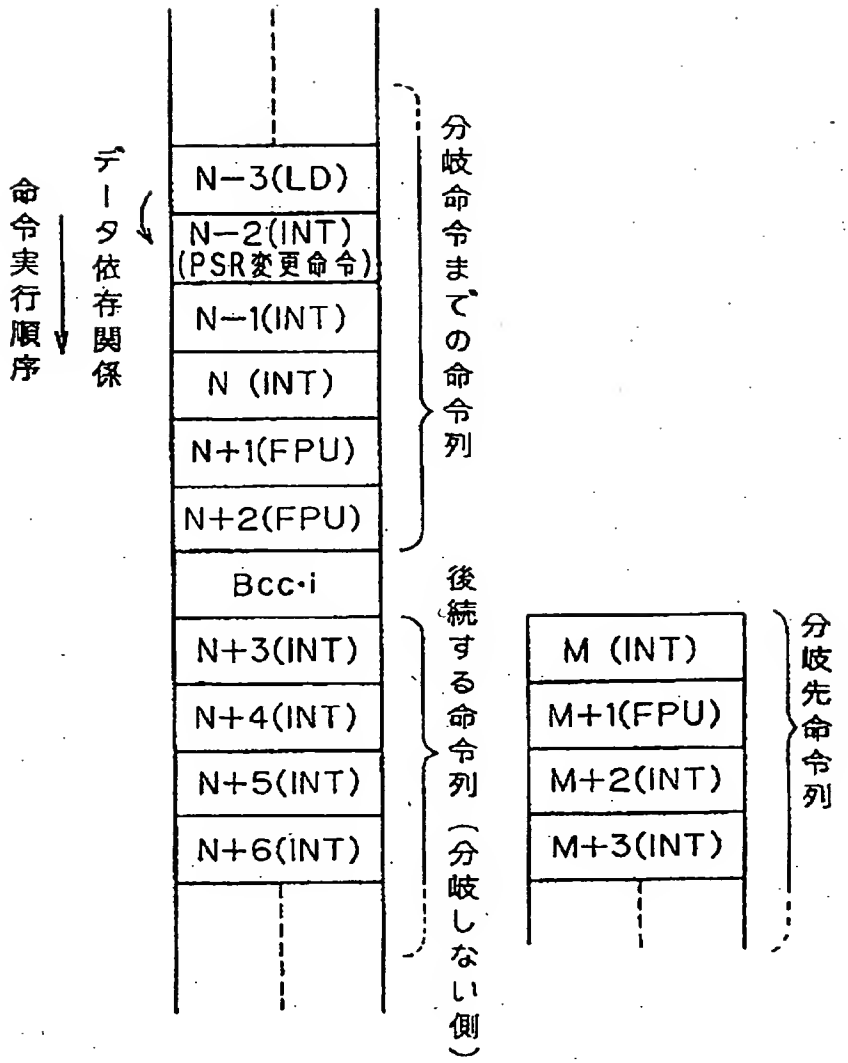
命令フェッチ部の詳細ブロック図



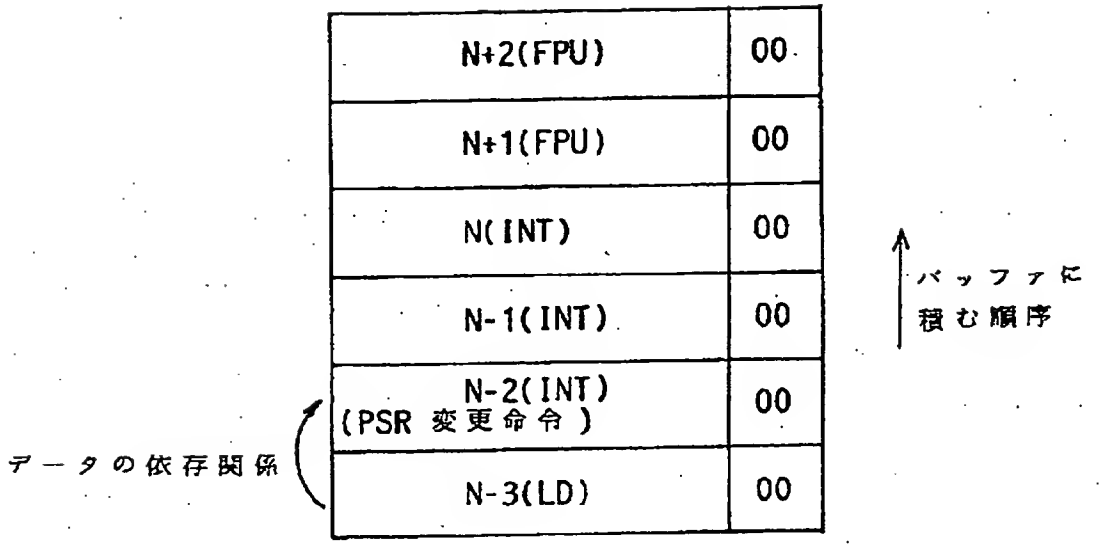
【図３】


セレクト
モード
33
32
モード付加
回路
ADDER
Reg.
"01" "10"
転送制御
回路
31
セレクタ５
の出力
分岐命令
検出回路
命令解読回路
命令発行回路
34
34a
34b
34c
34d
34e
34f
DEC
39
39a
39b
39c
39d
SEL
演算ユニット
命令解読
バッファ６
35
ＰＳＲを
変更する命
令を検出
40
命令発行
制御回路
初期モード
投機実行種類
36
データ依存
関係
37
スコアボード
管理回路
スコア
ボード２０
38
演算ユニット
管理テーブル１２
空き状態
検知

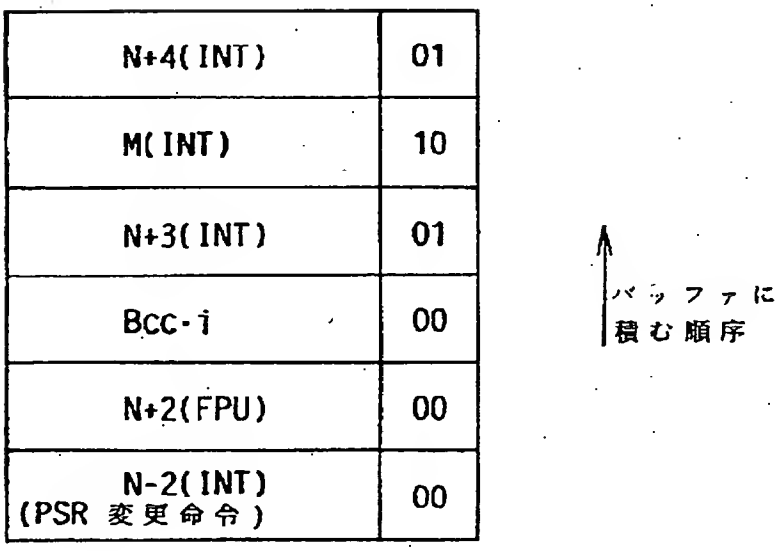
【図５】


N−3(LD)
N−2(INT)
(PSR変更命令)
N−1(INT)
N (INT)
N+1(FPU)
N+2(FPU)
Bcc·i
N+3(INT)
N+4(INT)
N+5(INT)
N+6(INT)
M (INT)
M+1(FPU)
M+2(INT)
M+3(INT)
データ依存関係
命令実行順序
分岐命令までの命令列
後続する命令列（分岐しない側）
分岐先命令列

【図6】


N+2(FPU) 00
N+1(FPU) 00
N(INT) 00
N-1(INT) 00
N-2(INT) (PSR 変更命令) 00
N-3(LD) 00
データの依存関係
バッファに積む順序

【図7】


N+4(INT) 01
M(INT) 10
N+3(INT) 01
Bcc-i 00
N+2(FPU) 00
N-2(INT) (PSR 変更命令) 00
バッファに積む順序

【図8】

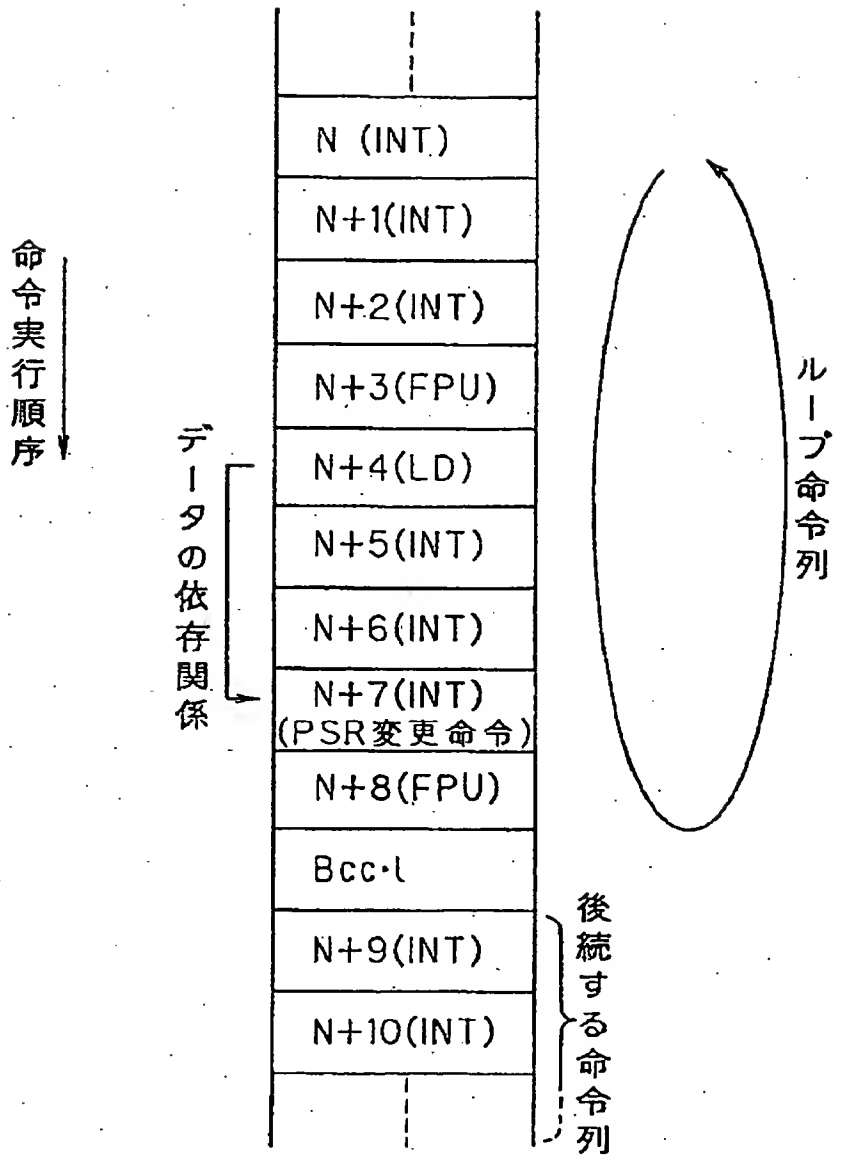
N+6(INT)	01
M+2(INT)	10
N+5(INT)	01
M+1(FPU)	10
N+4(INT)	01
N-2(INT) (PSR変更命令)	00

↑ バッファに積む順序

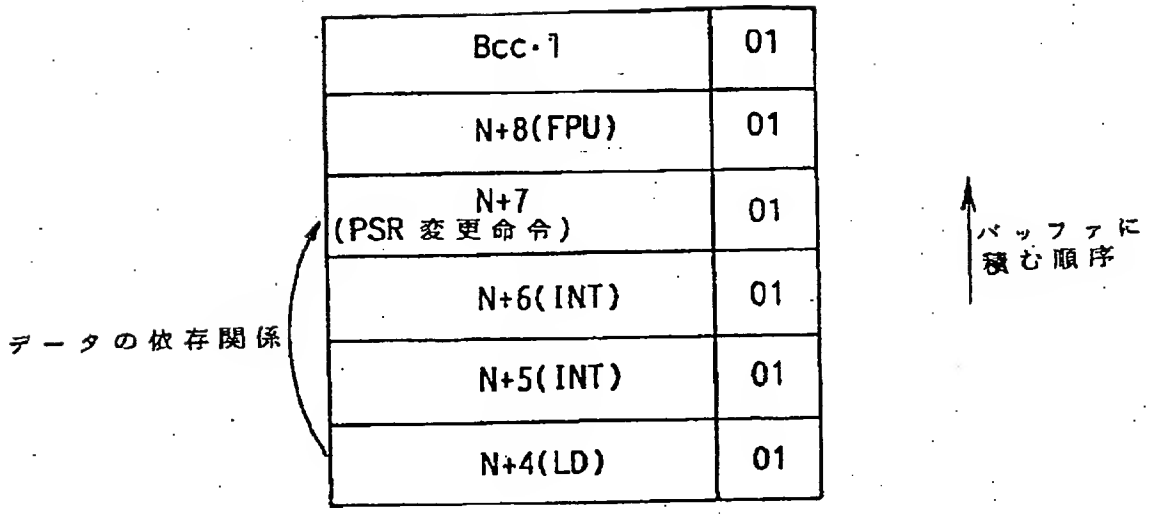
【図9】

モード	レジスタ番号	演算結果
01	(N+3)の演算結果を書き込むレジスタ番号	(N+3)の演算結果
10	(M) 〃	(M) 〃
01	(N+4) 〃	(N+4) 〃
10	(M+1) 〃	(M+1) 〃
01	(N+5) 〃	(N+5) 〃
10	(M+2) 〃	(M+2) 〃

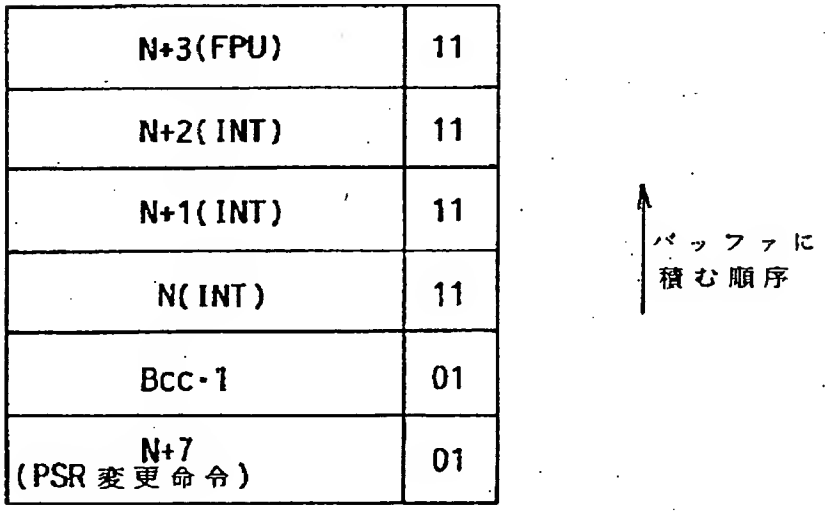
【図10】


命令実行順序
データの依存関係
N (INT)
N+1(INT)
N+2(INT)
N+3(FPU)
N+4(LD)
N+5(INT)
N+6(INT)
N+7(INT)
(PSR変更命令)
N+8(FPU)
Bcc·l
N+9(INT)
N+10(INT)
後続する命令列
ループ命令列

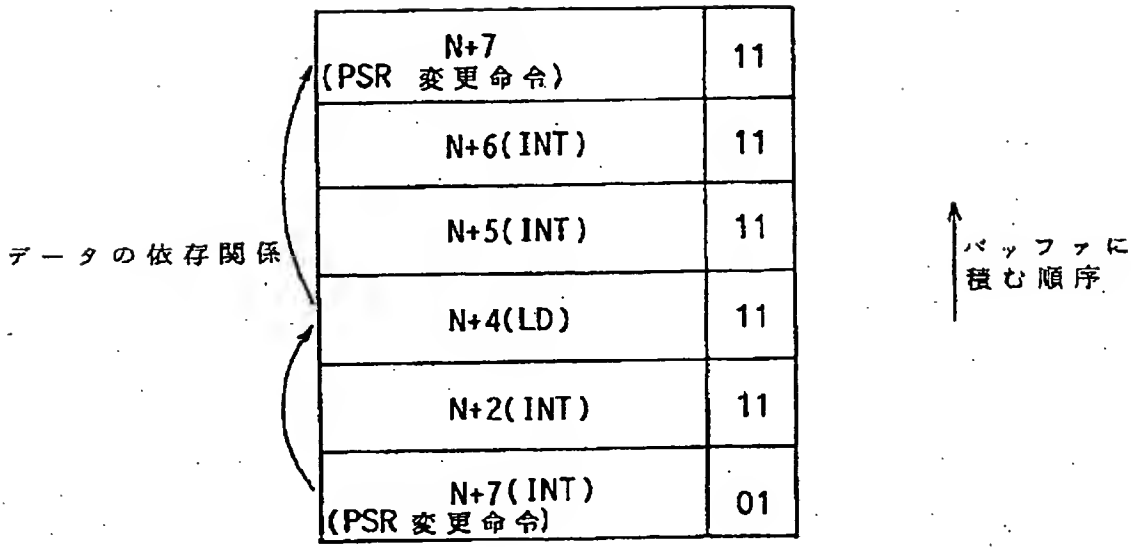
【図11】


Bcc・1
01
N+8(FPU)
01
N+7
(PSR 変更命令)
01
N+6(INT)
01
N+5(INT)
01
N+4(LD)
01
データの依存関係
バッファに
積む順序

【図12】


N+3(FPU)
11
N+2(INT)
11
N+1(INT)
11
N(INT)
11
Bcc・1
01
N+7
(PSR 変更命令)
01
バッファに
積む順序

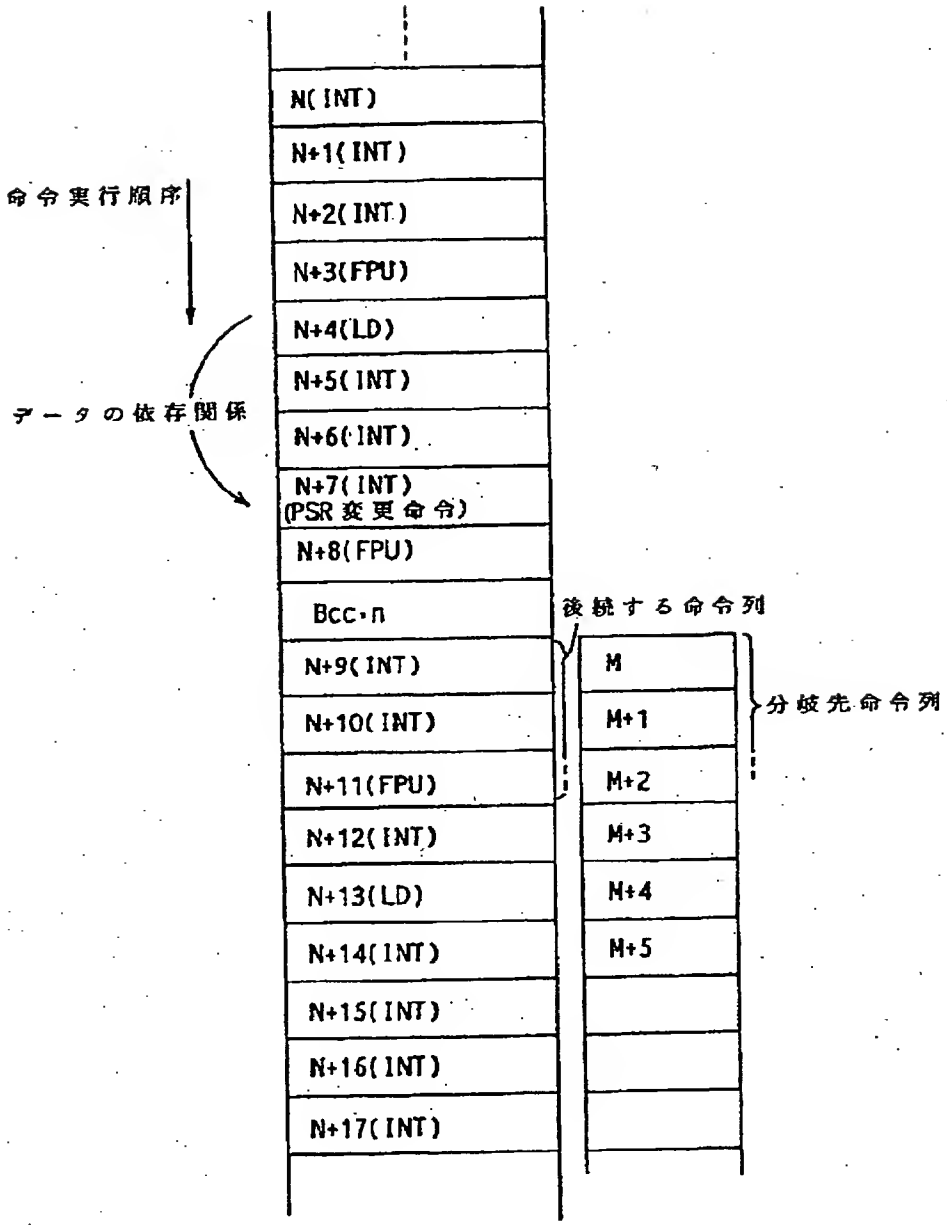
【図１３】



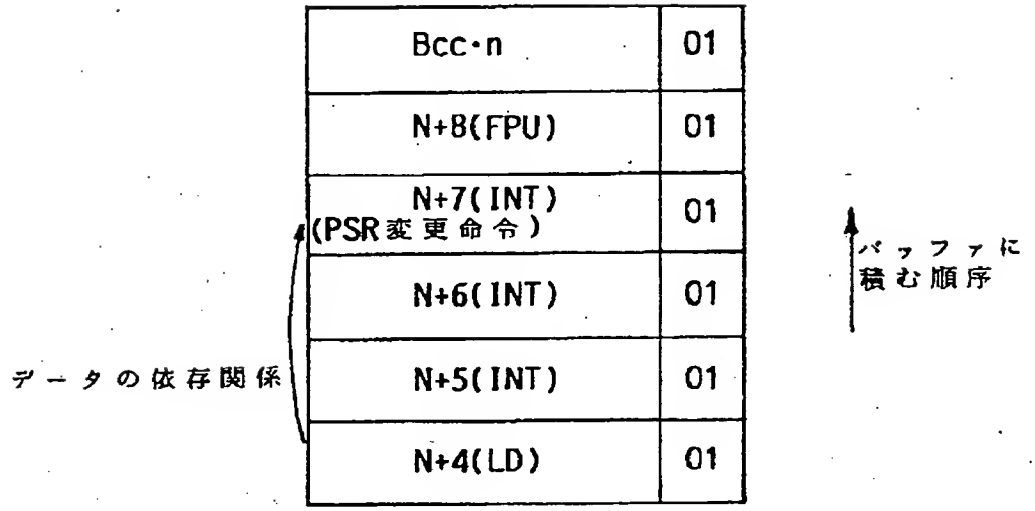
【図１４】

モード	レジスタ番号	演算結果
11	(N)の演算結果を書き込むレジスタ番号	(N)の演算結果
11	(N+1) 〃	(N+1) 〃
11	(N+3) 〃	(N+3) 〃
11	(N+2) 〃	(N+2) 〃

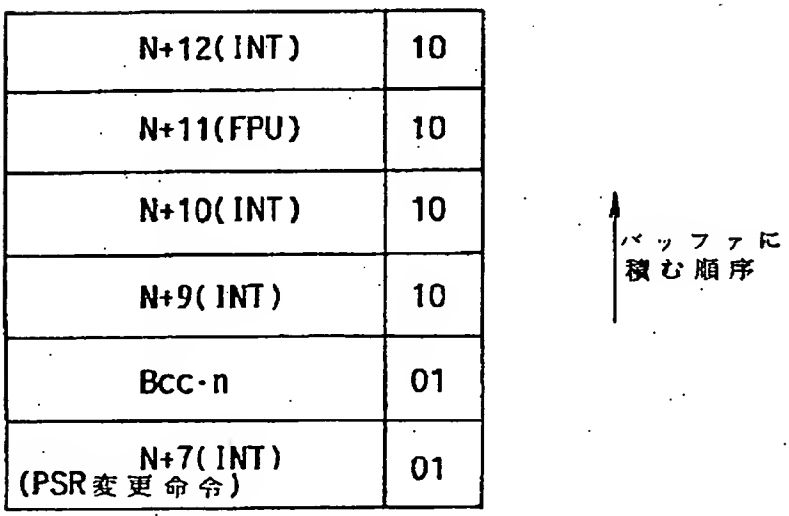
【図15】


命令実行順序
データの依存関係
N(INT)
N+1(INT)
N+2(INT)
N+3(FPU)
N+4(LD)
N+5(INT)
N+6(INT)
N+7(INT)
(PSR変更命令)
N+8(FPU)
Bcc・n
N+9(INT)
N+10(INT)
N+11(FPU)
N+12(INT)
N+13(LD)
N+14(INT)
N+15(INT)
N+16(INT)
N+17(INT)
後続する命令列
M
M+1
M+2
M+3
M+4
M+5
分岐先命令列

【図16】


Bcc・n
01
N+8(FPU)
01
N+7(INT)
(PSR変更命令)
01
N+6(INT)
01
N+5(INT)
01
N+4(LD)
01
データの依存関係
バッファに
積む順序

【図17】


N+12(INT)
10
N+11(FPU)
10
N+10(INT)
10
N+9(INT)
10
Bcc・n
01
N+7(INT)
(PSR変更命令)
01
バッファに
積む順序

【図18】

N+16(INT)	10
N+15(INT)	10
N+14(INT)	10
N+13(LD)	10
N+12(INT)	10
N+7(INT) (PSR 変更命令)	01

バッファに積む順序

【図19】

モード	レジスタ番号	演算結果
10	(N+9)の演算結果を書き込むレジスタ番号	(N+9)の演算結果
10	(N+10) 〃	(N+10) 〃
10	(N+11) 〃	(N+11) 〃
10	(N+12) 〃	(N+12) 〃
10	(N+13) 〃	(N+13) 〃